university of groningen

faculty of science and engineering

computing science

SC@RUG 2018 proceedings

# 15th SC@RUG 2017-2018

Rein Smedinga, Michael Biehl (editors)

www.rug.nl/research/jbi

# SC@RUG 2018 proceedings

Rein Smedinga<br>
Michael Biehl<br>
editors

2018<br>
Groningen

ISBN (e-pub pdf): 978-94-034-0736-4
ISBN (book): 978-94-034-0737-1
Publisher: Bibliotheek der R.U.
Title: 15th SC@RUG proceedings 2017-2018
Computing Science, University of Groningen
NUR-code: 980

# About SC@RUG 2018

## Introduction

SC@RUG (or student colloquium in full) is a course that master students in computing science follow in the first year of their master study at the University of Groningen.

SC@RUG was organized as a conference for the fifteenth time in the academic year 2017-2018. Students wrote a paper, participated in the review process, gave a presentation and chaired a session during the conference.

The organizers Rein Smedinga and Michael Biehl would like to thank all colleagues who cooperated in this SC@RUG by suggesting sets of papers to be used by the students and by being expert reviewers during the review process. They also would like to thank Femke Kramer and Lemke Kraan for giving additional lectures and Agnes Engbersen for her very inspiring workshops on presentation techniques and speech skills.

## Organizational matters

SC@RUG 2018 was organized as follows:
Students were expected to work in teams of two. The student teams could choose between different sets of papers, that were made available through the digital learning environment of the university, *Nestor*. Each set of papers consisted of about three papers about the same subject (within Computing Science). Some sets of papers contained conflicting opinions. Students were instructed to write a survey paper about the given subject including the different approaches discussed in the papers. They should compare the theory in each of the papers in the set and draw their own conclusions, potentially based on additional research of their own.

After submission of the papers, each student was assigned one paper to review using a standard review form. The staff member who had provided the set of papers was also asked to fill in such a form. Thus, each paper was reviewed three times (twice by peer reviewers and once by the expert reviewer). Each review form was made available to the authors through *Nestor*.

All papers could be rewritten and resubmitted, independent of the conclusions from the review. After resubmission each reviewer was asked to re-review the same paper and to conclude whether the paper had improved. Rereviewers could accept or reject a paper. All accepted papers[1] can be found in these proceedings.

In her lectures about communication in science, Femke Kramer explained how researchers communicate their findings during conferences by delivering a compelling storyline supported with cleverly designed graphics. Lectures on how to write a paper and on scientific integrity were given by Michael Biehl and a workshop on reviewing was offered by Lemke Kraan.

Agnes Engbersen gave workshops on presentation techniques and speech skills that were very well appreciated by the participants. She used the two minute madness presentation (see further on) as a starting point for improvements.

Rein Smedinga was the overall coordinator, took care of the administration and served as the main manager of *Nestor*.

Students were asked to give a short presentation halfway through the period. The aim of this so-called two-minute madness was to advertise the full presentation and at the same time offer the speakers the opportunity to practice speaking in front of an audience.

The actual conference was organized by the students themselves. In fact half of the group was asked to fully organize the day (i.e., prepare the time tables, invite people, look for sponsoring and a keynote speaker, create a website, etc.). The other half acted as a chair and discussion leader during one of the presentations.

Students were graded on the writing process, the review process and on the presentation. Writing and rewriting accounted for 35% (here we used the grades given by the reviewers), the review process itself for 15% and the presentation for 50% (including 10% for being a chair or discussion leader during the conference and another 10% for the 2 minute madness presentation). For the grading of the presentations we used the assessments from the audience and calculated the average of these.

The gradings of the draft and final paper were weighted marks of the review of the corresponding staff member (50%) and the two students reviews (25% each).

On 6 April 2018, the actual conference took place. Each paper was presented by both authors. We had a total of 19 student presentations this day.

In this edition of SC@RUG students were videotaped during their 2 minute madness presentation and during the conference itself using the video recording facilities of the University. The recordings were published on *Nestor* for self reflection. Both recordings however, had technical issues.

[1] this year, all papers were accepted

### Sponsoring

The student organizers invited one keynote speaker, Jeroen Vlek of *Anchormen*. The company sponsored the event by providing lunch and coffee. After sessions drinks were sponsored by *BelSimpel*.
Hence, we are very grateful to

- Anchormen
- BelSimpel

for sponsoring this event.

### Thanks

We could not have achieved the ambitious goals of this course without the invaluable help of the following expert reviewers:

- Vasilios Andrikopoulos
- George Azzopardi
- Michael Biehl
- Kerstin Bunte
- Laura Fiorini
- Jiri Kosinka
- Maria Leyva
- Jorge A. Perez
- Gerard Renardel
- Brian Setz
- André Sobiecki
- Nicola Striscuiglio
- Michael Wilkinson

and all other staff members who provided topics and provided sets of papers.
Also, the organizers would like to thank the *Graduate school of Science* for making it possible to publish these proceedings and sponsoring the awards for best presentations and best paper for this conference.

Rein Smedinga
Michael Biehl

Since the tenth SC@RUG in 2013 we added a new element: the awards for best presentation, best paper and best 2 minute madness.

**Best 2 minute madness presentation awards**

**2018**
Marc Babtist and Sebastian Wehkamp:
*Face Recognition from Low Resolution Images: A Comparative Study*
**2017**
Stephanie Arevalo Arboleda and Ankita Dewan
*Unveiling storytelling and visualization of data*
**2016**
Michel Medema and Thomas Hoeksema
*Implementing Human-Centered Design in Resource Management Systems*
**2015**
Diederik Greveling and Michael LeKander
*Comparing adaptive gradient descent learning rate methods*
**2014**
Arjen Zijlstra and Marc Holterman
*Tracking communities in dynamic social networks*
**2013**
Robert Witte and Christiaan Arnoldus
*Heterogeneous CPU-GPU task scheduling*

**Best presentation awards**

**2018**
Tinco Boekestijn and Roel Visser
*A comparison of vision-based biometric analysis methods*
**2017**
Siebert Looije and Jos van de Wolfshaar
*Stochastic Gradient Optimization: Adam and Eve*
**2016**
Sebastiaan van Loon and Jelle van Wezel
*A Comparison of Two Methods for Accumulating Distance Metrics Used in Distance Based Classifiers*
**and**
Michel Medema and Thomas Hoeksema
*Providing Guidelines for Human-Centred Design in Resource Management Systems*
**2015**
Diederik Greveling and Michael LeKander
*Comparing adaptive gradient descent learning rate methods*
**and**
Johannes Kruiger and Maarten Terpstra
*Hooking up forces to produce aesthetically pleasing graph layouts*
**2014**
Diederik Lemkes and Laurence de Jong
*Pyschopathology network analysis*
**2013**
Jelle Nauta and Sander Feringa
*Image inpainting*

**Best paper awards**

**2018**
Erik Bijl and Emilio Oldenziel:
*A comparison of ensemble methods: AdaBoost and random forests*
**2017**
Michiel Straat and Jorrit Oosterhof
*Segmentation of blood vessels in retinal fundus images*
**2016**
Ynte Tijsma and Jeroen Brandsma
*A Comparison of Context-Aware Power Management Systems*
**2015**
Jasper de Boer and Mathieu Kalksma
*Choosing between optical flow algorithms for UAV position change measurement*
**2014**
Lukas de Boer and Jan Veldthuis
*A review of seamless image cloning techniques*
**2013**
Harm de Vries and Herbert Kruitbosch
*Verification of SAX assumption: time series values are distributed normally*

# Contents

# A Review of Shape Enhancement Techniques

Patrick Vogel, Sietze Houwink

**Abstract**—The appearance of objects that have highly reflective metallic or painted finishes is primarily defined by the reflections of other objects. These reflections can be optimized to provide a better look to an object. For this purpose, there are several techniques available. In this paper, three of those techniques will be summarized. Apart from the summary, the applications of the technique will be discussed, including a number of cases in which the technique doesn't bring the desired effect.

The following methods that can be used in practice to obtain an improved reflection line distribution with respect to conventional methods will be introduced, explained and summarized. The first method substitutes the geometry of a three-sided cubic Bézier patch for the triangle's flat geometry, and a quadratically varying normal for shading, in order to improve the visual quality of existing triangle-based art. The second method uses a purely bi-cubic construction to irregular quad layouts that satisfies the highlight-line criterion of class A surfacing, thereby generalizing on the conventional method for regular quad layouts. The third method uses reflection lines, that capture many essential aspects of reflection distortion, for interactive surface interrogation and semi-automated surface improvement.

Once every technique is summarized and discussed, a comparison is provided. This comparison results in a new perspective on the techniques, and when they can be used. In order to help the reader with understanding this comparison, a decision tree is provided. This paper ends with a discussion of how the decision tree should be interpreted.

**Index Terms**— Computer Graphics, Reflection lines, Surfaces, Surface Shape Optimization

## 1 Introduction

The appearance of objects that have highly reflective metallic or painted finishes is primarily defined by the reflection of other objects. Reflections of a set of long linear parallel light sources can be thought of as a special type of reflected environment, capturing the distortion introduced by an object for a particular direction of features in the environment. For example, horizontal and vertical lines are most common in urban and indoor environments, and it makes sense to use these directions for surface optimization. These reflection lines can be optimized to provide a better look to the object (Fig. 1).

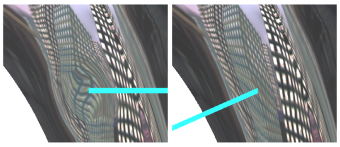


Fig. 1: An example of reflection line optimization. Taken from [3].

A reflection line depends on the location and the normal direction of a surface point, and is defined as follows. A reflection line [1, 4, 5, 6] on a surface $\mathbf{x}$ is defined by an eye point $\mathbf{e_p}$, the light plane, and a line $\mathbf{l}$ in the light plane. Considering $\mathbf{x}$ as a mirror, the reflection line on $\mathbf{x}$ is defined as the mirror image of $\mathbf{l}$ on $\mathbf{x}$ while looking from $\mathbf{e_p}$ (Fig. 2).

Conventional shape optimization methods typically require information about neighboring faces, which are therefore computationally expensive. The method in [8] as summarized in Sec. 2.1 does not require additional data beyond the position and normal data of the considered triangle, but is still able to soften triangle creases and improve the visual appeal by generating smoother silhouettes and better shading.

'Class A surface' is a term in the automotive design industry for describing spline surfaces with aesthetic, non-oscillating highlight lines. Conventional bi-3 constructions around irregular points, that generate a finite number of polynomial pieces by exploiting the freedom to reparameterize, have to date failed the restriction of non-oscillating highlight lines. The method in [3] as summarized in Sec. 2.2 does meet this requirement by allowing a slight mismatch of normals below the accepted tolerance.

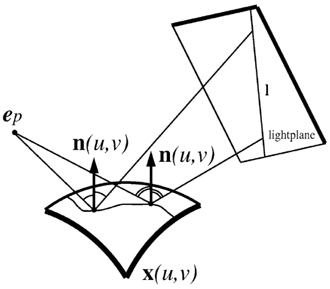


Fig. 2: Definition of a reflection line. Taken from [6].

The controls of a shape have an indirect effect on the reflective surface. The method in [7] as summarized in Sec. 2.3 formulates the surface editing problem as an optimization problem, as specified by the designer.

In Sec. 3, a decision tree is constructed from the three methods in Sec. 2. Sec. 4 provides a discussion of the mentioned methods, in terms of their results and their applications.

## 2 Methods

This section summarizes three methods that can be used in practise to obtain an improved reflection line distribution with respect to conventional methods, as described in [8, 3, 7].

### 2.1 Point-Normal Triangles

The method in [8] introduces the concept of curved point-normal triangles (PN Triangles). This is an inexpensive means to improve the visual quality of existing triangle-based art, providing smoother silhouettes, better shading, and more organic shapes (see Fig. 3).

A curved point-normal triangle replaces a flat triangle by a curved shape that is retriangulated into a desired number of flat subtriangles. At least cubic geometry variation and quadratic normal variation are

- *Patrick Vogel (p.p.vogel@student.rug.nl) and Sietze Houwink (s.g.houwink@student.rug.nl) are Msc Computing Science students at the University of Groningen.*

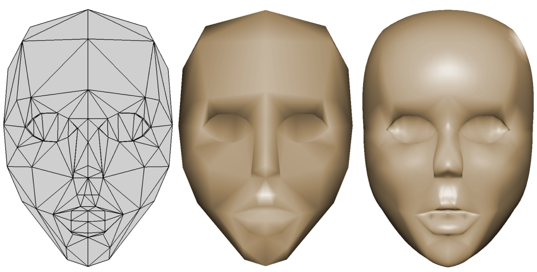


Fig. 3: (a) Input triangulation, (b) Gouraud shading (c) Curved point-normal triangles. Taken from [8].

required to capture inflections implied by the triangle positional and normal data.

The geometry is defined by a cubic Bézier patch (Eq. 1, Fig. 4), matching the points and normals of the vertices of the flat triangle, which influences the object silhouette.

$$b(u,v) = \sum_{i+j+k=3} b_{ijk} \frac{3!}{i!j!k!} u^i v^i w^k, \quad w = 1-u-v, \quad u,v,w, \geq 0 \quad (1)$$

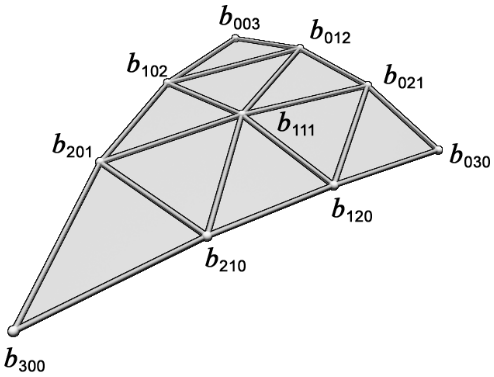


Fig. 4: The control points of the geometry. Vertex coefficients: $b_{300}$, $b_{030}$, $b_{003}$. Tangent coefficients: $b_{210}$, $b_{120}$, $b_{021}$, $b_{012}$, $b_{102}$, $b_{201}$. Center coefficient: $b_{111}$. Taken from [8].

The vertex normals are defined by an independently specified linear or quadratic Bézier interpolant (Eq. 2, Fig. 5), which influences the whole mesh.

$$n(u,v) = \sum_{i+j+k=2} n_{ijk} u^i v^i w^k, \quad w = 1-u-v, \quad u,v,w, \geq 0 \qquad (2)$$

Given the points and normals of the vertices of a flat triangle (Fig. 6), the geometry coefficients are obtained by the following construction. Explicit formulas for the coefficients are provided in [8].

1. Initialize the coefficients uniformly over the flat triangle, that is at positions $(iP_1 + jP_2 + kP_3)/3$.

2. Project each tangent coefficient into the tangent plane defined by the normal of the closest corner (Fig. 7).

3. Translate the center coefficient by 1.5 times the vector between the average of the tangent points, and the initial position of the center.

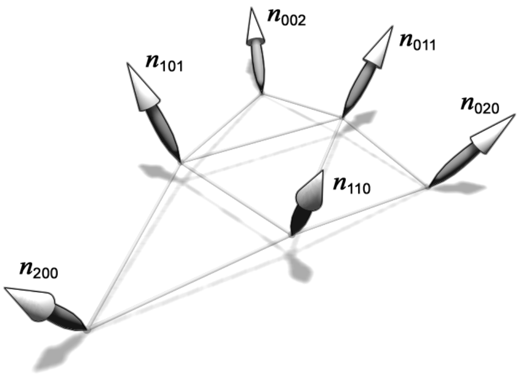


Fig. 5: The control points of the normals. Taken from [8].

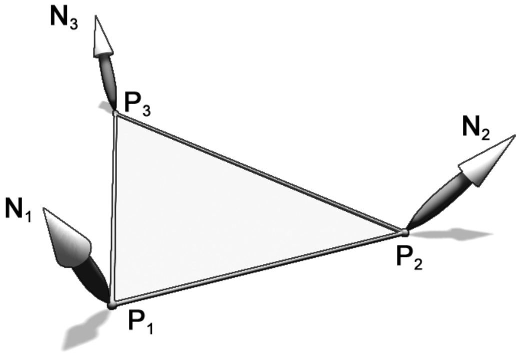


Fig. 6: Points $P_i$ and normals $N_i$. Taken from [8].

The particular choice of coefficients as proposed in [8] keeps the curved patch provably close to the flat triangle. This is stated in Theorem 1.

**Theorem 1** *[8] Let L be the length of the longest triangle edge. The tangent coefficients are within a distance L/6 from the flat triangle, and the center coefficient is within a distance L/4 from the flat triangle. The curved-point normal triangles do not usually join with tangent continuity except at the corners.*

To capture inflections as in Fig. 8, the mid-edge coefficient for the quadratic map $n$ is approximated by the average of the end-normals reflected across the plane perpendicular to the edge (Fig. 9). Explicit formulas for the coefficients are provided in [8].

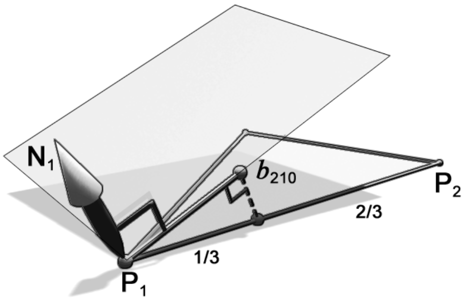


Fig. 7: Construction of a tangent coefficient. Taken from [8].

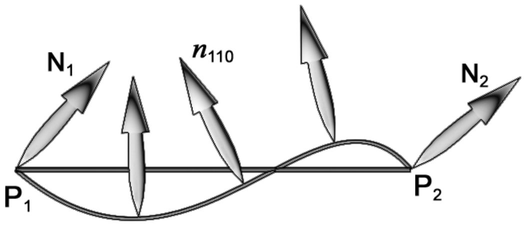


Fig. 8: Quadratic interpolation of the normals at the endpoints. Taken from [8].

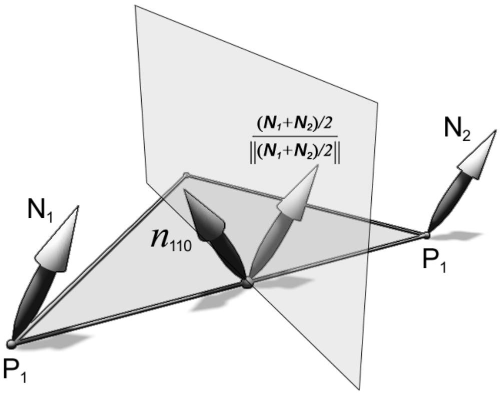


Fig. 9: Construction of the mid-edge normal coefficient. Taken from [8].

Vertices on a sharp or crease edge have two distinct normals, resulting in cracks or gaps in the surface, that cannot be avoided with entirely local information. To support edges of this kind, a software preprocessing step adds a rim of small triangles along edges intended to be sharp (Fig. 10).

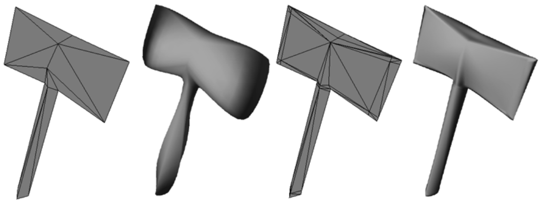


Fig. 10: Sharpening by adding a seam of small triangles. Taken from [8].

The overall rendering performance is limited by the bandwidth of the bus communicating with the GPU, rather than the processing power available for the geometry in the GPU. If the processor is sufficiently fast and the bus is busy, curved point-normal triangles render at the same speed as flat triangles.

Curved point-normal triangles can furthermore be used as a form of geometry compression, by representing meshes by higher order surface primitives. This reduces both bus bandwidth and memory usage significantly.

### 2.2 Class A Surfaces

The second method that we would like to discuss in this paper, is the method introduced in [3]. This method is designed for improving ‘Class A surfaces’, which is a term in the automotive design industry for describing spline surfaces with aesthetic, non-oscillating highlight lines [3]. The method works by obtaining an improved reflection line distribution near irregular points with respect to conventional methods. Class A highlight lines aim for a normal mismatch along the cap boundaries that is below the industry-accepted tolerance of a tenth of a degree. The underlying mesh can be constructed by tensor-product B-splines of degree bi-3 (Fig. 11), with boundaries aligned with feature curves. For tensor-product $4 \times 4$ sub-nets (with all valence $n = 4$ nodes), class A surfaces can be obtained directly, as the formulas for B-spline to Bernstein-Bézier form conversion can be applied directly. For sub-nets with a single interior node of valance $n \neq 4$ (Fig. 12), a surface within the accepted tolerance of class A surfaces can be approximated by transforming an existing high-quality higher-degree surface to a degree bi-3 surface cap, by the methods described in [3], thereby sacrificing formal smoothness and requiring a slight mismatch of normals on the surface cap boundary.

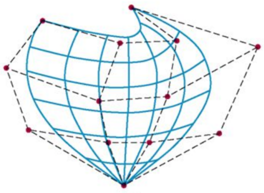


Fig. 11: Example of tensor-product B-splines of degree bi-3. Taken from [2].

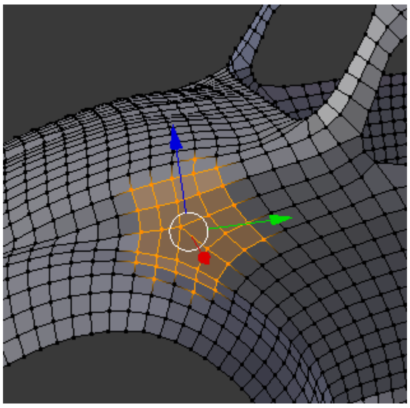


Fig. 12: Input net. Taken from [3].

Surfaces with boundaries are not considered, and it is assumed that each quad has only one node of valence $n \neq 4$, otherwise a Catmull-Clark subdivision step must be applied. For each valence $n$, the $n$-sided surface caps are linear combinations of the seven basic functions $h_l$ (Fig. 13).

The surface cap is fully defined by the nodes marked as bullets in Fig. 14. The B-spline to Bernstein-Bézier form conversion of the surrounding sub-nets yields boundary conditions that are used to define a degree bi-5 surface cap (Fig. 15). The Bernstein-Bézier coefficients indicated by red dots imply a well-defined curvature at the irregular point. The construction in Fig. 16 approximates the degree bi-5 surface cap by a degree bi-3 surface cap. The internal cap transitions are adjusted to be $G^1$ while leaving the cap boundaries $C^0$ (Fig. 17).

For a complex input, a comparison of the bi-3 surface and its bi-5 guide is presented in Fig. 18, to reveal the otherwise very small differences. Only the change in the Gauss curvature shading is visible.

Typically one Catmull-Clark refinement step improves the distribution of highlight lines and reduces the normal mismatch, but subsequent steps result in the resolution of different highlight lines coming together late and bunching up at the irregular point, obtaining non-class A highlight lines.

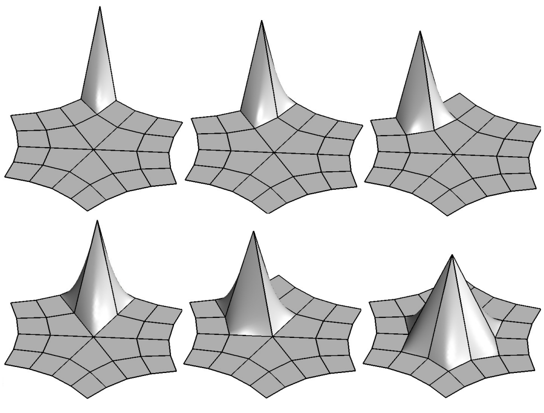


Fig. 13: Basic functions $h_l$. The upper row shows $h_1$ through $h_3$. The bottom row shows $h_5$ through $h_7$. $h_4$ is not shown as this is symmetric to $h_2$ across the diagonal of the same sector. Taken from [3].

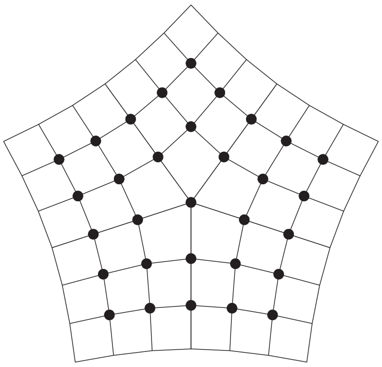


Fig. 14: The control points of the input net marked by bullet points. Taken from [3].

### 2.3 Reflection Functionals

The controls of a shape, which have an indirect effect on the reflection line distribution, can be automatically adjusted by optimization methods that minimize the deviation from a desired reflection line distribution, as specified by the designer. Such methods allow the designer to smooth and warp reflection lines, change reflection line density, and create surfaces with a desired reflection line distribution (Fig. 19).

A numerical technique that offers interactive performance is described in [7]. Consider the system presented in Fig. 20. Both the viewer and the light sources are located at infinity. The view direction is defined by the unit length vector **v**. The light sources are defined to be parallel to the unit length vector **a**, presented as lines on the bottom half of a cylinder. The normalized projection of **v** to the plane perpendicular to **a** defines $\mathbf{v}_a$. The normalized projection of **v** to the plane perpendicular to **v** defines $\mathbf{a}_v$. The cross product of **a** and $\mathbf{v}_a$ defines $\mathbf{a}_\perp$. The reflection of **v** at a point **p** on the surface with normal vector **n** defines the reflection vector **r**. The projection of **r** to the plane perpendicular to **a** defines **d**. The reflection line function $\theta$ is then defined by

$$\theta = \arctan(\mathbf{r}\cdot\mathbf{a}_\perp, \mathbf{r}\cdot\mathbf{v}_a) \tag{3}$$

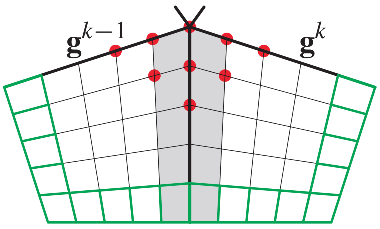


Fig. 15: Guide surface of degree bi-5. Taken from [3].

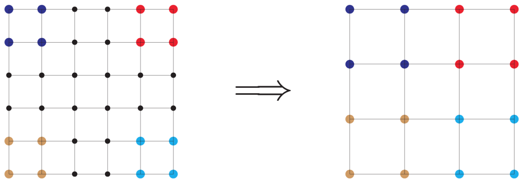


Fig. 16: Transformation from bi-5 to bi-3. Taken from [3].

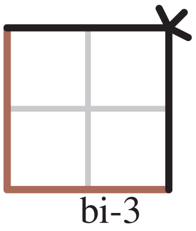


Fig. 17: Continuity properties of the surface cap. $G^1$ continuity for black edges, $C^1$ continuity for grey edges, $C^0$ continuity for brown edges. Taken from [3].

Consider the coordinate system $(\mathbf{a}_\perp, \mathbf{a}_v, \mathbf{v})$ that is aligned with the image plane, with coordinates $x, y, z$ along the axes. The local parametrization $z = f(x,y)$ of a mesh surface can be obtained by a simple linear transformation. All vertices close to the silhouette will be fixed to form boundary conditions for optimization regions. If as normal $\mathbf{n} = (f_x, f_y, 1)$ is taken, the expression for $\theta$ reduces to

$$\theta = \arctan\big(2f_x, -2f_y\sin\alpha + (1 - f_x^2 - f_y^2)\cos\alpha\big) \tag{4}$$

Several possible optimization problems are presented in [7]. We only summarize the minimization of the difference in line directions and density captured by the gradient of the reflection function, which is least restrictive in the required boundary conditions. Here the gradient of the reflection function is fitted to the gradient of the desired function.

$$\begin{aligned} &\text{minimize} \int_S (\nabla\theta - \nabla\theta^*)^2 dxdy \\ &\nabla\theta = \frac{r_2\nabla r_1 - r_1\nabla r_2}{r_1^2 + r_2^2} \\ &\theta|_{\partial S} = \theta_0 \\ &\frac{\partial}{\partial n}\theta|_{\partial S} = \varphi_0 \end{aligned} \tag{5}$$

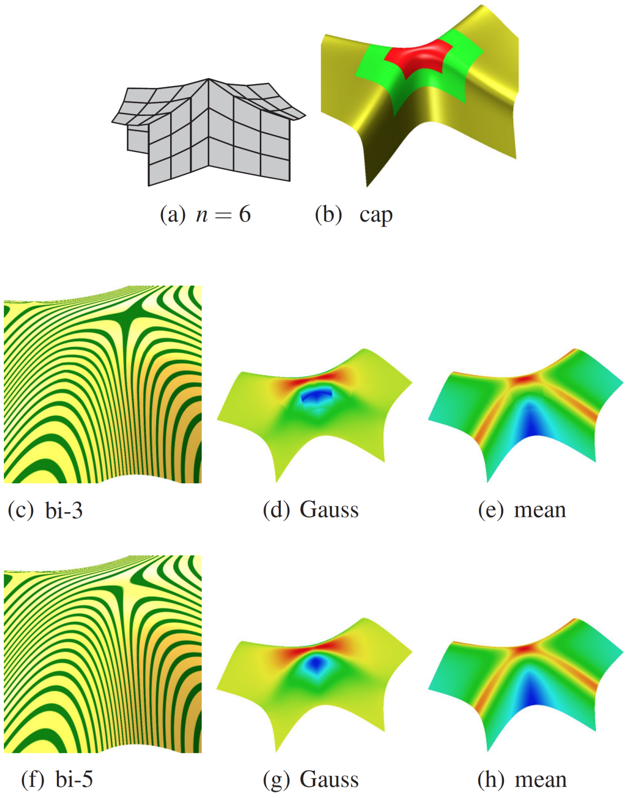
The corresponding Euler-Lagrange equation is fourth-order, and one can prescribe both Dirichlet and Neumann boundary conditions ensuring smooth transitions between the optimized patch and the surface.

To obtain the gradient and Hessian of $f$, triangle-centered discretization will be used, which assigns a single value of the gradient or Hessian to each face (Fig. 21). Let

$$f_T = \big(f(\mathbf{p}_1), f(\mathbf{p}_2), f(\mathbf{p}_3), f(\mathbf{q}_1), f(\mathbf{q}_2), f(\mathbf{q}_3)\big) \tag{6}$$

A discretization of the gradient, using standard piecewise linear continuous finite elements, yields

$$\nabla_{discr} f_T = \frac{1}{2A} \sum_{i=1,2,3} f(\mathbf{p}_i)\mathbf{t}_{ii} \tag{7}$$


(a) $n = 6$ (b) cap

(c) bi-3 (d) Gauss (e) mean

(f) bi-5 (g) Gauss (h) mean

Fig. 18: Comparison bi-3 surface and its bi-5 guide. Taken from [3].

A discretization of the Hessian, is approximated by a combination of two approaches. Triangle-averaged discretization yields

$$\mathrm{H}_{discr} f_T = \frac{1}{A}\left(\sum_{i,j,j\neq i}\frac{1}{A_j} f(\mathbf{q}_j)\mathbf{t}_{ii}\otimes\mathbf{t}_{ij} + \sum_{i}\frac{1}{A_i} f(\mathbf{p}_i)\mathbf{t}_{ii}\otimes\mathbf{t}_{ii}\right) \quad (8)$$

The discretization of the Hessian introduces mesh dependent errors (Fig. 22), which are fortunately low-frequency while high frequency errors would have the most effect on the results.

The alternative is quadratic interpolation discretization, where the coefficients of a quadratic function, matching the vertices $\mathbf{p}_i$ and $\mathbf{q}_j$ (Fig. 21), are used to estimate the Hessian. This approach is consistent whenever the quadratic function is defined, but is highly unreliable when six points of the stencil are close to a common conic. To solve this problem, triangle averaged discretization is used whenever the quadratic interpolation discretization is unstable. Hybrid discretization yields the best results (Fig. 22).

To obtain a better normal for a vertex $\mathbf{p}$, a local fit on the ring $N$ of triangles around $\mathbf{p}$, and all triangles edge adjacent to $N$, can be used.

## 3 Comparison

In Sec. 2 three existing methods that can be used to obtain a smoother shape by optimizing the reflection line distribution have been described. In this section a comparison of those methods is provided. This results in a decision tree.

Sec. 2.1 describes how the visual quality of triangle-based art can be improved by using Point-Normal Triangles. This method doesn't produce the desired effect when the triangles contain sharp edges. Point-Normal triangles provide a smoother, though not necessarily everywhere tangent continuous, silhouette and more organic shapes [8], as presented in Fig. 10.

Sec. 2.2 describes how an improved reflection line distribution can be obtained by optimizing 'Class A surfaces'. This methods works only on quads, but the resulting cap transitions are adjusted to be $G^1$ while leaving the cap boundaries $C^0$ [3].

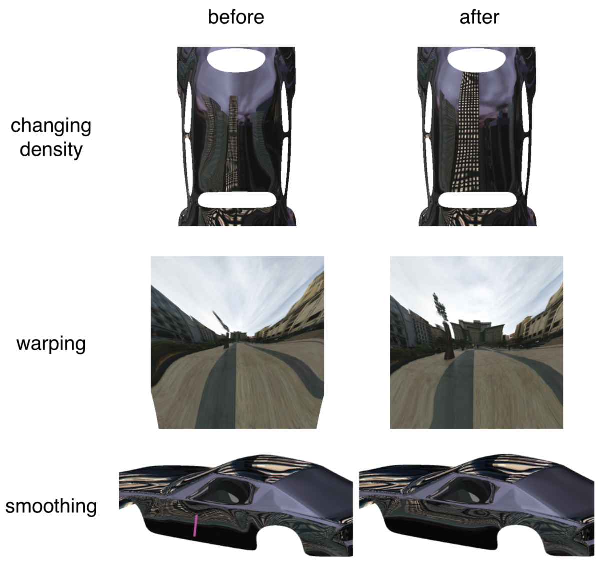


Fig. 19: Examples of reflection line optimization. Taken from [7].

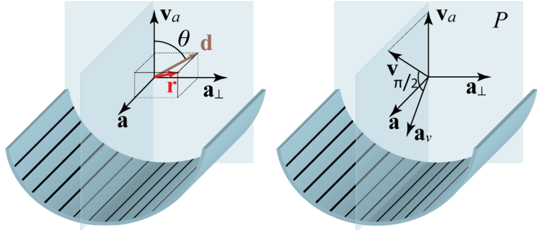


Fig. 20: System definition. Taken from [7].

Sec. 2.3 describes a method of how a surface with the desired reflection line distribution can be obtained in a semi automatic approach. One limitation of the proposed approach is that the vertices of the mesh move only in the direction perpendicular to the image plane. This means that small scale surface details which make the projection to the image plane not one-to-one cannot be removed. Although it can be applied to large perturbations, the technique is best suited for smaller adjustments of surfaces that are already relatively smooth [7].

All three methods have been combined into a simple decision tree. This tree can be found in Fig. 23 and presents the information of when each method can be applied.

## 4 Discussion

Our research aims to determine when and where one of the three methods can be used. The methods are described in Sec. 2 and a comparison of those methods is given in Sec. 3.

The purpose of this decision tree is to provide simplicity for the reader of this paper. When the reader deals first with those methods, it is often hard to understand them properly. The decision tree can help in understanding the purpose of each of the method. The decision tree only provides information when a method can possibly be used, but this doesn't apply in all cases. Note for example, that the method described in Sec. 2.1 can be used if the mesh consists of triangles, but this method doesn't produce the desired effect when the shape contains sharp edges.

Apart from this decision tree, we would like to point out that some methods can be performed without human interaction, while the third method (Sec. 2.3) cannot, as it is semi-automated. Therefore, it is not possible to implement the decision tree in an algorithm, as it some-

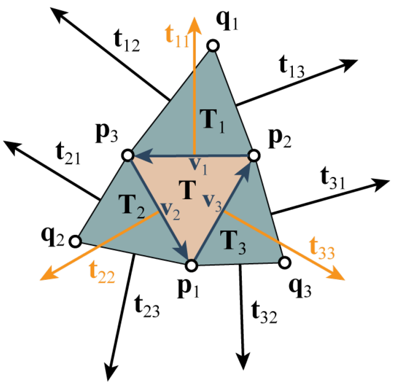


Fig. 21: Used to obtain gradient and Hessian. The vectors $t_{ij}$ are side perpendiculars, which have the same length as the sides. Taken from [7].

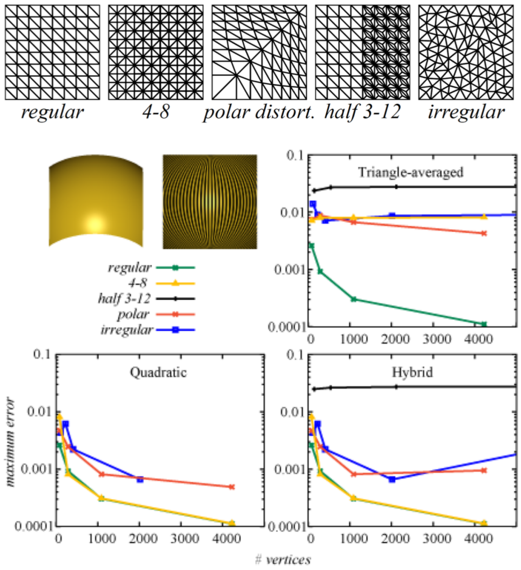


Fig. 22: Mesh types and convergence experiments. Taken from [7].

times needs to communicate with the designer of the shape.

## 5 Conclusion

In this paper, we have discussed three methods that can be used to obtain a smoother shape by optimizing the reflection line distribution. The first method substitutes the geometry of a three-sided cubic Bézier patch for the triangle's flat geometry, and a quadratically varying normal for shading, in order to improve the visual quality of existing triangle-based art. The second method uses a purely bi-cubic construction to irregular quad layouts that satisfies the highlight-line criterion of class A surfacing, thereby generalizing on the conventional method for regular quad layouts. The third method uses reflection lines, that capture many essential aspects of reflection distortion, for interactive surface interrogation and semi-automated surface improvement. In order to better understand each method for the reader of this paper, we have described and summarized the methods. Furthermore, a comparison of the methods is provided. This comparison results in a decision tree, which can be found in Fig. 23.

### Acknowledgements

The authors wish to thank J. Kosinka for his useful feedback in the draft version of this paper.

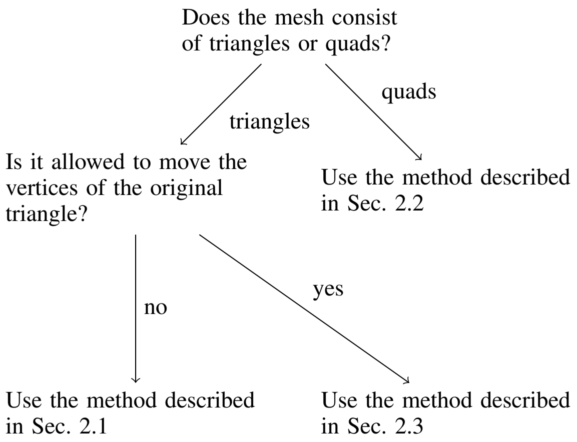


Fig. 23: Decision tree for the comparison of the three methods

## References

[1] H. Hagen, S. Hahmann, T. Schreiber, Y. Nakajima, B. Wordenweber, and P. Hollemann-Grundstedt. Surface interrogation algorithms. *IEEE Computer Graphics and Applications*, (5):53–60, 1992.

[2] Jehee Lee, Seoul National University. Splines, Bezier Surfaces (slide 26). Available at: http://slideplayer.com/slide/4635199/. Accessed 24 March 2018.

[3] K. Karčiauskas and J. Peters. Can bi-cubic surfaces be class A? *Comput. Graph. Forum*, 34(5):229–238, Aug. 2015.

[4] E. Kaufmann and R. Klass. Smoothing surfaces using reflection lines for families of splines. *Computer-Aided Design*, 20(6):312–316, 1988.

[5] R. Klass. Correction of local surface irregularities using reflection lines. *Computer-Aided Design*, 12(2):73–77, 1980.

[6] H. Theisel. Are isophotes and reflection lines the same? *Computer Aided Geometric Design*, 18(7):711 – 722, 2001. Pierre Bzier.

[7] E. Tosun, Y. I. Gingold, J. Reisman, and D. Zorin. Shape optimization using reflection lines. In *Proceedings of the Fifth Eurographics Symposium on Geometry Processing*, SGP '07, pages 193–202, Aire-la-Ville, Switzerland, Switzerland, 2007. Eurographics Association.

[8] A. Vlachos, J. Peters, C. Boyd, and J. L. Mitchell. Curved PN triangles. In *Proceedings of the 2001 Symposium on Interactive 3D Graphics*, I3D '01, pages 159–166, New York, NY, USA, 2001. ACM.

# A review of reference architectures enabling the Internet of Things

Loran Oosterhaven and Johan de Jager

University of Groningen, Faculty of Science and Engineering

**Abstract**—Most IoT-devices currently available are developed by different vendors and have no clear standards or protocols for communication. Each device differs in capability regarding hardware such as internet connectivity, GPS, vision and other types of sensors and hardware. Hence, the desired level of connectivity between these devices cannot be achieved in the current state of the Internet of Things without introducing some form of a generic architecture or framework. These frameworks must allow heterogeneous devices and services to be able to communicate with each other. Extensive research has been conducted in the last decade and many different architectures and frameworks have been introduced to tackle these challenges. In our paper several of these architectures and frameworks are discussed. For each proposed architecture or framework we looked at how well they are able to solve the introduced challenges. We also look at both their strengths, weaknesses and what makes them unique. Finally, we evaluate how well each framework handles security and privacy concerns.

**Index Terms**—Internet of Things, Distributed Systems, Reference Architectures, Heterogeneous Networks

## 1 Introduction

In recent years the Internet of Things (IoT) has become a well discussed topic of innovation in our society. The increasing availability of the internet world-wide and decreasing costs of sensors and actuators in combination with the advances in cloud computing has created the potential to achieve true connectivity. In the context of IoT, true connectivity means that people and objects can be connected at anytime, anyplace with anything and anyone using any network and any service [7].

IoT can potentially make our lives more efficient, less stressful and substantially safer. For example, it can drastically change our health care system by utilizing wearable devices that can detect a host of health problems and react accordingly. Moreover, truly energy efficient smart homes can be created by automatically controlling lights, air conditioning and other electronic devices based on human activity. Besides the just given examples many other groundbreaking applications for the Internet of Things exist and will certainty be developed in the near future.

Achieving this common goal of connectivity unfortunately comes with many unsolved challenges of which some can potentially be solved by the introduction of some architecture or framework. Many frameworks have been proposed throughout years of research [6, 7, 2]. An important question that remains to be asked is which of these frameworks will eventually result in the highest level of connectivity and will best solve the known challenges of IoT.

## 2 Challenges in Internet of Things

We will briefly discuss the most significant challenges of IoT and explain what implications they could have for the Internet of Things.

### 2.1 Global standards for architecture and interconnection

One of these challenges is caused by the large number of device and service vendors involved in the Internet of Things. Lack of global standards are a large cause of incompatibilities between privately developed platforms and solutions [6]. This issue of heterogeneity is current often solved by the introduction of human-readable protocols of web services which can result in non-negligible overhead.

### 2.2 Scalability, performance and latency requirements

The effectiveness of the Internet of Things also heavily depends on the combined performance and scalability of all of these devices and services. Hence, issues regarding performance and scalability form another major challenge for the goal of true connectivity. The process of collecting, integrating, aggregating and processing huge amounts of data originating from many devices in order to transform them in the knowledge required by these smart services is very demanding. Besides the issues regarding performance and scalability we also have requirements in terms of the permitted amount of latency (e.g. self-driving vehicles need sensors with low latency for obvious safety considerations). Therefore, when introducing any additional layers in a smart network to address the just mentioned heterogeneity issues, one has to keep this requirement in mind and thus strive for low latency and high performance.

### 2.3 Usability by non-expert users

The Internet of Things should provide simplicity to non-experts users. As the existing IoT platforms do not offer off-the-shelf applications, users need to build applications from scratch or write code by using the provided tools and manuals for a specific device [7]. For many existing IoT platforms, programming expertise is required and therefore reusing existing components is difficult resulting in increased costs.

### 2.4 Security and privacy concerns

As soon as the impact of Internet of Things on our lives increases, the importance of security and privacy also increases. Securing devices that store privacy sensitive information about our health and current physical state is critical. For example, consider a burglar knowing whether or not we are home, sleeping or on vacation has drastic consequences for the safety of our household effects. Another example would be that an attacker is able to maliciously send fake requests to our smart home and control the gas stove in our kitchen. Hence, it is vital to have a reliable and well-balanced security framework present in the network [3]. It should be able to prevent unauthorized access and should guarantee that our home privacy persists.

## 3 Structure

The rest of the paper is organized as follows. In section 4, various reference architectures are discussed. For each of the reference architectures the design choices and important features are summarized. In section 5 all of the reference architectures are compared and evaluated. Their strengths and weaknesses are determined along with how well they address the challenges discussed in section 2. Finally, we present our conclusion in section 6.

## 4 Reference architectures

The reference architectures considered in this article have many similarities. All of the reference architectures make use of the ontology

- *Loran Oosterhaven, E-mail: l.oosterhaven@student.rug.nl.*
- *Johan de Jager, E-mail: j.m.de.jager.2@student.rug.nl.*

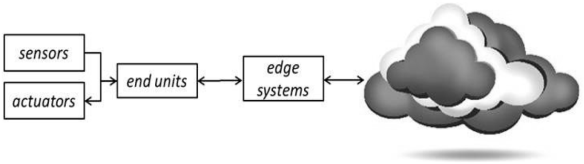


Fig. 1. The IoT systems and the connection to cloud.

method to better solve the heterogeneity issues between various devices and sensors. In computer science an ontology is a formal naming and definition of the interrelationships, types and properties of the entities that exist in a particular domain of interest [1]. In this situation the domain of interest is limited to all the objects making up the Internet of Things with the addition of some objects regarding security. One could think of radio frequency identification (RFID) tags, actuators, smart phones and other sensors as being entities in the ontologies for IoT. Besides using the ontology method, all discussed architectures also incorporate some form of cloud computing or decentralized computing. Managing many heterogeneous devices and performing cross-platform harmonization of their data is only feasible by relying on the advantages of cloud computing (i.e. virtually unlimited data storage and computing resources) [6].

### 4.1 Architecture 1: Smart gateway framework for IoT services

Smart gateway frameworks bridge the semantic gaps between raw data coming from sensors and the high level interpretation of this data. The main purpose of the framework described in this framework is to provide a high level abstraction of connected devices that make them easy and intuitive to access. Conversion of high level concepts like ”temperature at a given location” is translated to ”data from sensor X” by the gateway framework instead of the application using the data. Typical IoT can be depicted as in Figure 1. Sensors and actuators are connected to end units and are connected to the Internet through edge systems. These edge systems can also act as firewalls, implement routing and act as proxies. The framework allows application users to register new devices and make them discoverable. Events can be triggered based on sensor data given certain rules and the framework can manage privacy and security policies.

#### 4.1.1 Overall Architecture

The overall architecture of the smart gateway framework is shown in figure 2.

The Data Manager is responsible for managing the raw data coming from the sensors. A part of the Data Manager is the Time Manager: this component timestamps each sensor value. Currently, two storage strategies are supported: in-memory and cloud storage. In-memory storage works with a simple ring buffer that maintains the most recent sensor values. The cloud storage strategy provides a theoretically infinite storage space.

The Device Manager manages access to the sensors and devices. Also, it keeps track of all instantiated and deployed devices. Every virtual and physical device communicates with other devices and Managers through the D-bus. The Notification Agent is a sub-component of the Device Manager. It is implemented as a virtual device and allows for a simple publish-subscribe framework for events. New event types can be created and added to the notification agent.

The Context Manager maintains information about the semantics of the environment managed by the gateway. This is done by using ontologies. The Context Manager consists of two components: Device Catalog and Ontology Manager. The Device Catalog is a database of devices that can be queried and the Ontology Manager makes all the objects and their relations available to the framework in a uniform and flexible manner. The Rule Manager is the brain of the gateway. Periodically it evaluates rules and performs actions if the conditions defined by the rules are met.

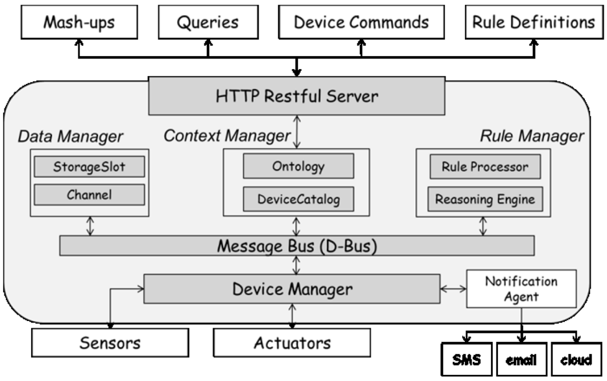


Fig. 2. The overall architecture of the smart IoT gateway.

The HTTP REST Server provides a simple uniform and easy way to access all the functionality of the framework. For example, the API can be used to find certain devices that match certain criterion using an SQL-like language. Also, it can be used to perform actions on certain devices, like turning on the light in a room or create new actions when a criterion is matched.

### 4.2 Architecture 2: Global generic architecture for the future IoT (GGIoT)

In this paper an architecture is proposed to meet the 6A Connectivity of the Internet of Things. The goal of 6A Connectivity is to allow people and objects to be connected anytime, anyplace, with anything and anyone, using any path/network and any service. The GGIoT architecture meets several architectural requirements described in the upcoming sections.

#### 4.2.1 Architectural requirements

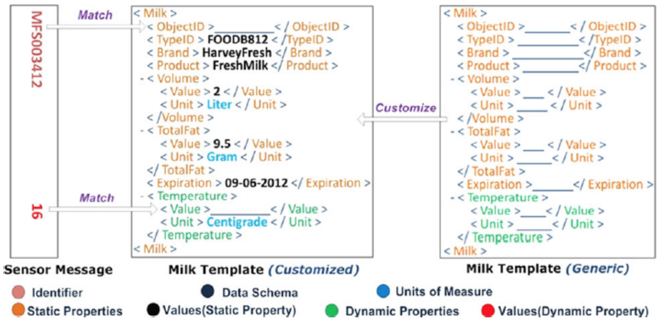
Objects need to be able to interoperate and work reliably with other objects. Interoperability is defined as the capability of integrating heterogeneous devices, networks, systems, services, APIs and data representation across domains and systems [5]. Standard web protocols can be used to request data from IP-enabled devices, but the overhead from these protocols is non-negligible.

The service oriented architecture is an architecture style independent of specific technologies and products. As opposed to Web Services, binary protocols are allowed for communication in component-based middleware. This allows for a lower data rate and less overhead.

Existing IoT platforms do not provide reusable and modularized services due to the diversity of devices. This makes building and consuming new services difficult and increases development costs. In the GGIoT, physical objects and services are virtualized as primitive middleware components. By loose coupling, virtual objects and services can be individually added, removed and reconfigured.

Objects need to communicate with multiple other objects at the same time. Self-driving vehicles for example need to communicate with cars nearby and traffic signals. In the GGIoT, a virtual object or service can be connected on demand to multiple services and objects at the same time. In real time, devices can be removed and added dynamically to the network. Therefore, the GGIoT needs to allow dynamic coordination of the interactions between virtual objects and services. Objects and services are dynamically reconfigured and terminated to meet specific application needs.

When physical objects move between spaces they will come into connection with new unknown objects. To prevent meaningless connections and wasting resources, objects should only connect to and communicate with certain types of objects. In the GGIoT, a distributed proxy is used to coordinate communication. If there is no service that can be provided between two devices, the proxy will not allow the connection.



```
< Milk >
  < ObjectID >__________< / ObjectID >
  < TypeID > FOODB812 < /TypeID >
  < Brand > HarveyFresh </ Brand >
  < Product > FreshMilk </ Product >
 - < Volume >
    < Value > 2 < / Value >
    < Unit > Liter < / Unit >
  < /Volume >
 - < TotalFat >
    < Value > 9.5 < / Value >
    < Unit > Gram < / Unit >
  </ TotalFat >
  < Expiration > 09-06-2012 </ Expiration >
 - < Temperature >
    < Value >__________< / Value >
    < Unit > Centigrade < / Unit >
  </ Temperature >
< Milk >
```

```
< Milk >
  < ObjectID >________< / ObjectID >
  < TypeID > __________< /TypeID >
  < Brand > ___________</ Brand >
  < Product > _________</ Product >
 - < Volume >
    < Value > ___ < / Value >
    < Unit > _____ < / Unit >
  < /Volume >
 - < TotalFat >
    < Value > ___ < / Value >
    < Unit > _____< / Unit >
  </ TotalFat >
  < Expiration >______</ Expiration >
 - < Temperature >
    < Value >____< / Value >
    < Unit > _____< / Unit >
  </ Temperature >
< Milk >
```

Fig. 3. Object description of a message.

Most existing IoT systems use centralized remote web servers to process the sensor data and expose URIs to access this sensor data through RESTful APIs. Aside from overhead from the web protocols, accessing sensor data through many networks and servers significantly increases latency.

Simplifying deployment is important to promote the use of IoT. To simplify deployment, GGIoT allows third-party users to reuse existing services. Devices can be connected using a plug-and-play mode which requires no programming.

### 4.2.2 Object description

To reduce resource usage, devices transmit only two values: the identifier of the device and the current state of the sensor. Meanings of the values are determined by the backend system. An example of this is shown in Figure 3 for a pack of milk. The sensor transmits only the object ID (MFS003412) and the state of the sensor (16). The milk template describes the meaning of the value 16: a temperature of 16 degrees Celcius. The template also specifies some static values: the grams of fat, total volume and other information about the product. This template is a customized version of the generic milk template; the generic template does not provide values for the static properties. Moving the interpretation to the middleware tier significantly reduces the size of the sensor messages. These templates can be modified, reused and shared to make the deployment of new devices easier.

### 4.2.3 Overall Architecture

The architecture consists of three tiers: perception tier, routing tier, middleware tier and the global management system (GMS). Figure 4 demonstrates this architecture.

The perception tier is the tier in which raw data is collected. It consists of all the sensors.

The routing tier is responsible for building communication channels between devices and the middleware in the proxies. This routing can be done by a variety of intermediate devices.

The middleware tier consists of distributed proxies. Each proxy consists of multiple components: identification system, look up system, database system, virtualization system, ontologies and the application system. The identification system is responsible for assigning and managing object IDs. The application system provides third parties with several development tools like template editing tools and tools to describe objects and services. The ontologies describe the relations between objects and services. The GMS is responsible for managing the ontology data and keeping it up to date. The lookup system enables discovery of virtual objects and services.

### 4.2.4 Building ontologies

The GGIoT consists of several ontologies: object, service and unit ontology. Respectively, they describe the relations between objects, services and units of measure. Similarly to how objects were defined, services can be defined by specifying input and output objects/services.

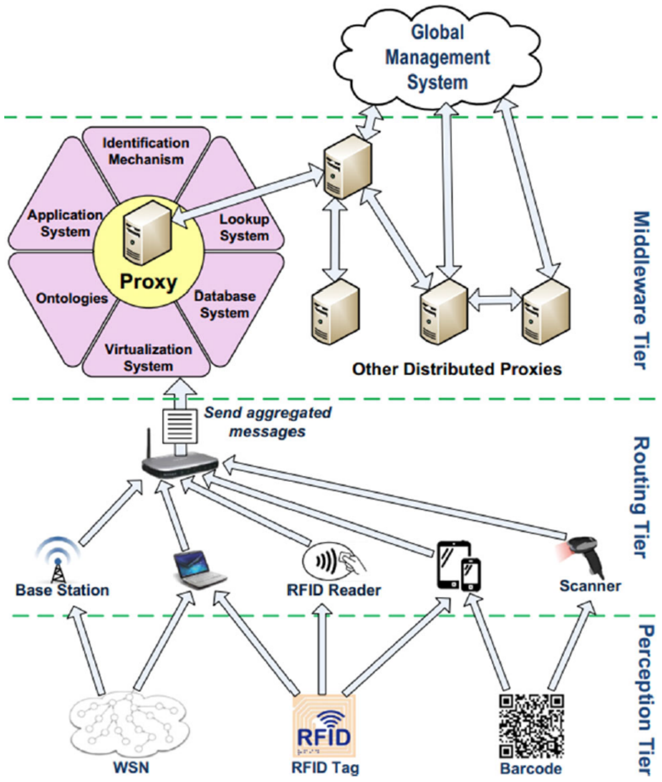


Fig. 4. Overall architecture of GGIoT.

## 4.3 Architecture 3: Multi-layer cloud architectural model

The multi-layer cloud architectural model proposed by Wang et al. [7] was based on the idea that given the many heterogeneous devices and available protocols and technologies in order to achieve cross-platform harmonization of their data it is only really feasible to rely on the virtually unlimited computing and storage resources provided by cloud computing. Furthermore, by shifting the computation of complex tasks towards the cloud instead of the sensors and actuator devices it decreases the computation and storage requirements on the actual devices. To tackle the heterogeneity issue the authors propose a strategy to build a public cloud on top of the private clouds of each device vendor to allow for a virtualized gateway for third-party applications.

### 4.3.1 Overall Architecture

The architecture for an IoT-based home consists of a layered scheme as presented in Figure 5. The bottom layers act as foundational support for the top layers. The middleware layer is introduced to hide the implementation details of the underlining technologies. The architecture is service-oriented (SOA) to integrate information and connect multiple devices from different vendors. Each vendor can have their own private cloud combined with their own access protocols and communication standards while the introduced public cloud provides a virtualized interface for third party access to home services.

The platform bus on the public cloud provides protocol conversion for all of the registered devices on the platform. Using this cloud oriented platform, different stakeholders, such as device vendors, government agencies and other third-party service providers can deploy a variety of applications using this approach while communicating with devices from different vendors having different protocols.

There are two scenarios to consider when a customer wants to manipulate a home device:

1. The target device is associated to the same private platform as the customers application.

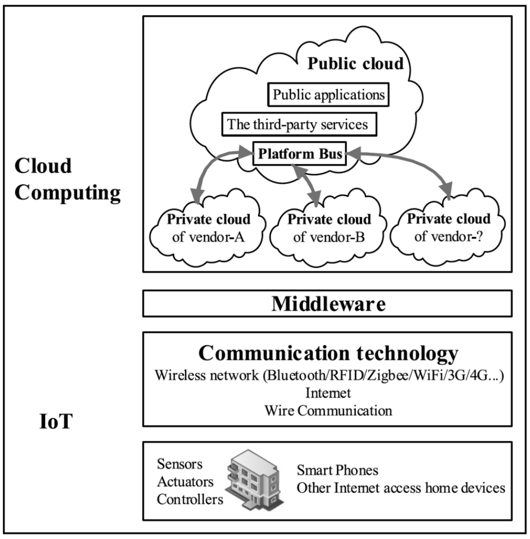


Fig. 5. Overall architecture of multi-layer cloud architectural model.

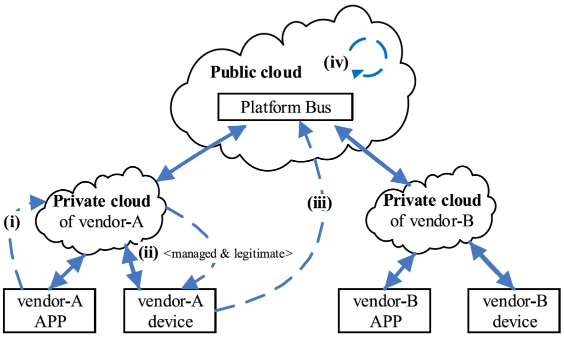


Fig. 6. Situation where target device and customer application are from the same vendor.

2. The target device is associated to a different private platform as the customers application.

The first scenario is schematically presented in Figure 6. Here a customer would use the vendor-specific application and send an operation command to the respective private cloud from the vendor. Next, the device ID will be checked by the private cloud. Since the device is managed by the same private platform the operation command will be immediately forwarded to the target device. After completing the request, the private platform will synchronize the target device status with the public cloud. Finally, the platform bus at the public cloud will synchronize the device status with all the other private platforms.

The second scenario is were the heterogeneity issue comes into play. Here the customer uses the application from one vendor and needs to send an operation command to a target device from a different vendor. The scenario is schematically presented in Figure 7. The first step of the process remains the same. However, here the private cloud of the application determines that target device is associated within this private platform. Hence, it forwards the operation command to the platform bus of the public cloud. The platform bus will now send the operation command to the correct private platform associated with the target device using its device ID. The operation command will now be completed by the private platform of the target device. This platform will now synchronize the device status with the public cloud using the platform bus. Finally, the platform bus synchronizes the device status in the entire cloud platform.

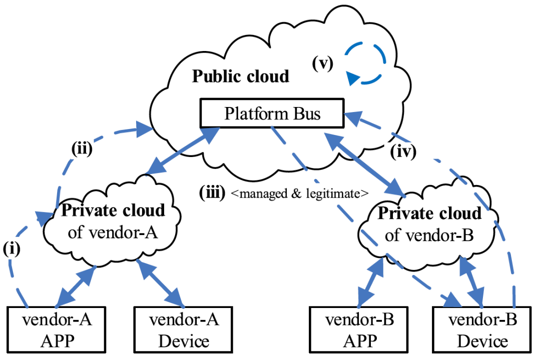


Fig. 7. Situation where target device and customer application are from different vendors.

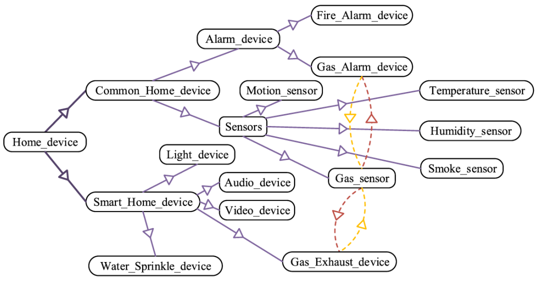


Fig. 8. The low-level concept of a home device..

### 4.3.2 Ontology-based modelling

Just as in the other architectures, ontology is being used to address data, knowledge and application heterogeneity. These ontologies are effectively deployed to describe and model available IoT devices in a generic way [4]. Hence resulting in a truly virtualized interface. Semantic Web Rule Language (SWRL) is being used to define the domain ontology and reasoning rules required for interaction and interoperations among the heterogeneous devices and services. As an exampl, a low-level concept for a home device is presented in Figure 8.

Using this virtualized representation of a home device, SWRL-based reasoning description for interactions and interoperations can be defined. An example of such reasoning for handling a fire alarm is presented in Figure 9. Here several conditions from varying sensors are being checked using logical operators. If all conditions are satisfied, the water device and fire alarm are being triggered. The benefits of this approach is that the rules are quiet easily readable, even for non-expert users and it can be applied to automate many processes within a smart environment.

### 4.3.3 Ontology-based security management

The architecture also uses ontology-based security management for interactions and interoperations between different sensors and applications. The security ontology of the architecture is presented in Figure 10. The ontology consists of two important classes. First the security objective, which indicates what kind of security is required (i.e.

$$
\begin{array}{l}
Location(?x) \\
\wedge((Smoke\_sensor(?sensor\_ID) \wedge atLocation(?sensor\_ID,?x) \wedge Environment\_smoke(?s,?x) \\
\wedge swrlb: greaterThan(?s, concentration\_threshold)) \vee (Temperature\_sensor(?sensor\_ID) \wedge \\
atLoaction(?sensor\ \ ID,?x) \wedge Environment\ \ temperature(?t,?x) \\
\wedge swrlb: greaterThan(?t, temperarure\ \ threshold))) \wedge (WaterDevice(?device\ \ ID) \wedge \\
atLoaction(?device\_ID,?x) \wedge hasFunction(?device\ \ ID, sprinkle)) \wedge (FireAlarmDevice(?device\_ID) \\
\wedge atLocation(?device\_ID,?x)) \rightarrow TriggerWaterDevice(?device\_ID) \wedge TriggerFireAlarmDevice(?device\_ID)
\end{array}
$$

Fig. 9. SWRL-based reasoning for handling a fire alarm.

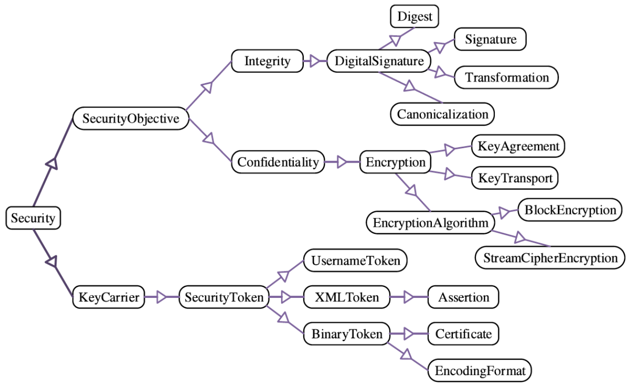


Fig. 10. The defined security ontology of the multi-layer cloud architecture.

```
<sec:AES-128>
    <sec:Token>
        <sec:Reference URI=”# X.509PKIPathToken ”/>
    </sec:Token>
    <sec:EncryptedParts>
        <sec:Body/>
    </sec:EncryptedParts>
</sec:AES-128>
```

Fig. 11. An example of an encryption assertion.

confidential data or integrity checking). For the integrity class a digital signature is required and several different hashing algorithms can be specified. For the confidential class different encryption algorithms and key transport algorithms can be specified. The key carrier class is used for carrying security keys. This is commonly done using tokens to hold keys outside or within a message.

Based on the security ontology just presented policies can be designed to indicate the abilities of interactions and interoperations. Both the customer and the service provider should define security policies in their interest. The final resulting security policy will then be the intersection of these policies. An policy consists of several assertions which represent security requirements. For example the whether or not a certain encryption algorithm has to be used. The architecture uses Web Ontology Language based operators to determine whether or not the assertion of one policy is compatible with assertions from another policy. An example assertion for encryption is presented in Figure 11.

## 5 Evaluation

To properly evaluate the discussed reference architectures, we will examine all the challenges of IoT mentioned in section 2 individually for each reference architecture. Arguments are given for each challenge why a certain solution by one architecture might be better than a solution provided by another architecture. At the end of this section, a table is presented where we try to objectively determine whether or not a reference architecture was able to properly solve a challenge. Evaluating a reference architecture for a challenge of IoT was naturally almost entirely based on qualitative research using the papers of the designers of each framework. However when quantitative data was available for a reference architecture it has been used to shape the evaluation. Each challenge will now be evaluated one by one.

### 5.1 Global standards for architecture and interconnection

All authors recognize that the most important problem to solve is heterogeneity. Components implement a common interface and then ontologies are defined to describe the relations between the components. Then, rules can be defined on these relations and components to trigger certain events, like firing alarms.

Table 1. Table of which architectures solve which challenges

| Architecture | 1 | 2 | 3 |
|---|---|---|---|
| Interconnection | X | X | X |
| Scalability |  | X | X |
| Usability |  | X | X |
| Security |  | X | X |

### 5.2 Scalability, performance and latency

The authors of the Smart Gateway architecture have not performed any tests for large numbers of devices on the network, but their evaluation for a small number of devices already shows rapid increase in response numbers as device actions increase. Clearly, the framework is not designed for connectivity on a global scale.

The GGIoT makes sure to limit bandwidth usage by having the sensors only transmit the bare minimum: the sensor ID and the sensor value. As no processing/preparing of the data is done on the devices with limited computing capability, latency stays low. This minimizes the chance of overloading the network with data. The distributed proxies make sure data processing and ontology management is decentralized, splitting workload across machines. This is very suitable for connectivity on a global scale. However, large scale tests have not been performed to confirm that this works on large systems.

### 5.3 Usability by non-expert users

The Smart Gateway does not attempt to make devices easy to integrate by non-expert users. To add new devices to the network, subclasses of the C++ class `DeviceBase` must be made. Clearly, this is not something non-expert users can do.

GGIoT makes this simpler by offering a global repository of templates that can be used. Users can easily create composite templates to combine multiple sensors into more complicated sensors.

Clearly the multi-layered cloud architecture provides the most elegant way for non-expert users to add new logic rules to the platform by introducing the use of SWRL-based reasoning descriptions which are more easy to understand. However, there is no repository of templates available for combined sensors as with GGIoT. A repository of pre-defined SWRL-based rules can of course be created in the future.

### 5.4 Security and privacy concerns

Only the multi-layer cloud architecture defines an ontology-based security management system which can be used to define rules regarding confidentiality and integrity of data. In their paper they discuss and explain how their system may be used to protect the platform against common attacks (e.g. man-in-the-middle attacks).

The Smart Gateway system uses gateways to control what data is published and to whom. This way information is not shared with unauthorized users.

GGIoT hides access to unauthorized parties by specifying ”communities” in templates. Only those communities mentioned can access the data of the given sensor. GGIoT claims improved privacy by omitting descriptions of the data. Surely this might make it slightly harder when data is intercepted but often it is still trivial to identify the meaning of the data by just looking at the data. For example, if we intercept values that fluctuate around the value 21, this could be the indoor temperature of someone’s house and most likely not the speed of a car, considering the variance of such data is generally larger.

We end our evaluation with Table 1, here we summarize which architecture solve which challenges. It is immediately visible that the first architecture is not able to address all challenges. While the later two architecture do solve all our determined challenges. In terms of scalability of GGIoT and the multi-layered cloud architecture it is hard to draw immediate conclusion as the tests from both authors are difficult to compare. However, the a solution for potential security issues has been addressed in a much more comprehensive way by the multi-layered cloud architecture.

## 6 Conclusion

In this paper several reference architectures have been discussed and evaluated. All architectures provide a solution for the main challenge of the current state of IoT, namely the heterogeneity issues caused by the many different vendors of devices. However, we can conclude that the solutions by some reference architecture deem to be more promising than the solutions of others. Architecture 3 solve this issue in the most elegant way. This is due to the introduction of an additional layer resulting in a public cloud communicating with vendor-specific private cloud as well as how straight forward defining reasoning rules is by using SWRL-based reasoning descriptions for interconnection and interaction between various sensors and devices. This is also why we believe it addresses the challenge of usability by non-expert users best. Since defining the logic and implications of certain events is easier to interpret than in the other reference architectures. As for the challenge regarding security and privacy, architecture 3 also most accurately overcomes the challenge. The options the architecture provides regarding encryption and digital signing are very dynamic and comprehensive using their ontology-based security management system. Hence, the security and privacy are simply much more advanced and easy to use than those provided by the other architectures. This can also be related to the fact that for architecture 3 security and privacy are described much more in detail. In conclusion we can thus say that architecture 3 seems to be the most promising reference architecture for the future. However, as the authors state themselves, future research is required before large-scale deployment can be performed.

It is important to mention that more reference architectures for IoT exist besides the ones discussed in this paper. Therefore, for further research is crucial to evaluate and compare more of these architectures. In the end, one of the largest challenges for any of the architectures is the adoption rate. Building and implementing a global IoT architecture with a high adoption rate will require a lot of hard work and collaboration between many groups such as academia, home device companies, law enforcement organizations, government authorities, standardization groups and cloud service providers [6].

## References

[1] T. R. Gruber. A translation approach to portable ontology specifications. *Knowledge Acquisition*, 5(2):199 – 220, 1993.

[2] Y. H. Lee and S. Nair. A smart gateway framework for iot services. In *2016 IEEE International Conference on Internet of Things (iThings) and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom) and IEEE Smart Data (SmartData)*, pages 107–114, Dec 2016.

[3] J. Li, Y. K. Li, X. Chen, P. P. C. Lee, and W. Lou. A hybrid cloud approach for secure authorized deduplication. *IEEE Transactions on Parallel and Distributed Systems*, 26(5):1206–1216, May 2015.

[4] C. Qu, F. Liu, and M. Tao. Ontologies for the transactions on iot. *Int. J. Distrib. Sen. Netw.*, 2015:3:3–3:3, Jan. 2015.

[5] Saint-Expury. Internet of things, strategic research roadmap. *Internet of Things Initiative, Surrey*, 2009.

[6] M. Tao, J. Zuo, Z. Liu, A. Castiglione, and F. Palmieri. Multi-layer cloud architectural model and ontology-based security service framework for iot-based smart homes. *Future Generation Computer Systems*, 78:1040 – 1051, 2018.

[7] W. Wang, K. Lee, and D. Murray. A global generic architecture for the future internet of things. *Service Oriented Computing and Applications*, 11(3):329–344, Sep 2017.

# Energy management optimization in energy hubs

E. Werkema, S.R. Boelkens, *Master students, University of Groningen*

**Abstract**—Renewable energy and electric vehicles have gained interest over the past decade, due to the decreasing fossil fuel reserves and increasing environmental problems. Energy management is changing due to the development in information technology towards a more cloud-based approach. In order to deal with this shift in energy management, the term ”Energy Hub” has been introduced. Energy hubs are decentralized systems in which incoming energy resources are distributed to local demand. In this paper an overview of the current modeling and optimization techniques in the scientific literature for energy management in energy hubs is presented. Energy hubs contain various energy storage, production, and conversion systems for different types of energy carriers. In such integrated systems the electrical and thermal loads pose uncertainties in modeling the energy carriers and has to be taken into account when optimizing the operational costs or energy loss. Several energy hub models and mathematical optimization techniques are introduced and discussed in this paper in context to their impact on reliability, scalability, and accuracy. It is shown that current energy hub models vary greatly and are therefore difficult to be compared in context to their applied optimization technique.

**Index Terms**—Energy Hub, Energy Management, Distributed Energy Resources, Mixed Integer Linear Programming (MILP), Mixed Integer Non-Linear Programming (MINLP).

## 1 Introduction

Reducing greenhouse gas emissions and growing energy demands are subjects gaining more attention in the fields of Electrical and Computer Engineering. The technical and organizational foundations of the energy industry are therefore gradually changing in order to cope with the increasing significant environmental issues and decreasing fossil fuel reserves. A possible concept that has been proposed to solve this problem is the ”energy internet” [10]. This concept proposes a new way of distributing and managing the energy resources in a decentralized, efficient, and reliable way. One of the important parts of this concept are the energy hubs that provide this decentralized behaviour. An energy hub is defined as a decentralized system in distributing incoming energy resources to local demand, by integrating generation conversion, and storage technologies [9].

An energy hub can be represented as an interface between different energy streams that have a varying load. From a systematic view this is represented by energy carriers as input and output. The internal system of an energy hub can consist of energy production mechanisms, energy storage, and/or energy conversion technologies. This systematic view is depicted in Figure 1.

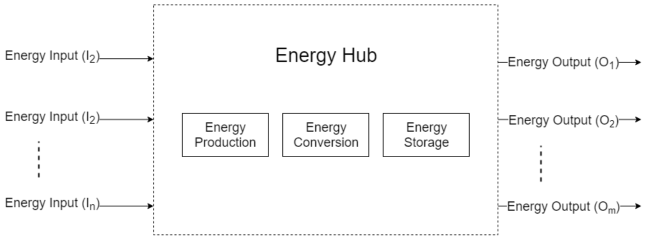


Fig. 1. Systematic view of an energy hub.

The U.S. and most countries within Europe have a liberalized energy market, and therefore have a varying energy price depending on the demand. Ideally in such a liberalized market you would like the supply and demand to correspond to each other. This reduces operational costs since energy loss is minimized. Energy hubs offer a great tool for simulating these varying demands and reflecting this on the supply by varying the loads on the systems. Resulting in an optimal price of energy to reduce total costs. In the case of a complete connected system, in which the devices are all connected to each other and customers can schedule the energy usage of their devices, the energy hubs can propose schedules for the devices to customers. These schedules are made based on the price for electricity at specific times. In order to have an efficient participation in the energy market, a demand response program (DRP) has been introduced to schedule this varying demand to face the uncertainties and variability associated with it. As defined by the U.S. Department of Energy, ”DR is the consumers capability to change their energy consumption pattern, in response to the price signal variations over time or to incentive payments offered by utilities in order to accomplish reasonable prices as well as system reliability during on peak periods” [1].

In the past decade hundreds of researchers investigated energy hubs in context to various energy input carriers and storage technologies [7]. Besides for example electricity from a power grid, natural gas is also widely used as energy input due to the recent developments in power-to-gas (P2G) and combined heat and power (CHP) technologies [9]. Also the development of concentrating solar power (CSP) has been taken into account in models. Another important influence on the future energy hubs is represented by the electrical vehicles (EV). Electrical vehicles use a significant amount of energy and are flexible in their charging schedule. They could even be used to discharge their energy in case of idle vehicles and high demand of the grid. Besides requiring energy for heating, houses can also require energy for cooling and pose another type of conversion in the system. This trade-off can be solved using underground water storage and could be used in the energy grid as an alternative energy storage [6].

This paper tries to focus on the computational point of view in context to energy hubs. More specifically it tries to answer the following research question:

> *What are the current modeling and optimization techniques in the scientific literature for energy management in energy hubs and how do these techniques impact the reliability, scalability and accuracy of the energy hub?*

Section 2 provides the definitions of energy hub models in context to their scale. The optimization problems and possible techniques for solving them are introduced in section 3. Section 4 presents the problems that arise when combining household devices, renewable energy generation systems and a demand response program to an energy hub. Consequently in section 5 these techniques are compared based on their impact on reliability, scalability, and accuracy of the energy hub. Finally a conclusion is made in section 7 between these techniques and gives an overview of how to apply them in various environments and what the impact will be.

*For information on obtaining reprints of this article, please send e-mail to: e.werkema@student.rug.nl.*

## 2 Energy Hub Models

Energy hubs can be represented as interfaces between energy consumption, production and transportation. The schematic view of an energy hub as depicted in Figure 1 consists of energy inputs and outputs. Within the energy hub energy can be converted or stored by using technologies such as CHP systems, transformers, and heat exchange technologies. Each of these systems again can have a number of inputs and outputs. Therefore the general mathematical definition of an energy hub can be represented by Equation 1 taken from the article from Alipour *et al.* [1].

$$\begin{bmatrix} L_\alpha \\ L_\beta \\ \vdots \\ L_\phi \end{bmatrix} = \begin{bmatrix} m_{\alpha,\alpha} & m_{\beta,\alpha} & \cdots & m_{\omega,\alpha} \\ m_{\alpha,\beta} & m_{\beta,\beta} & \cdots & m_{\omega,\beta} \\ \vdots & \vdots & \ddots & \vdots \\ m_{\alpha,\phi} & m_{\beta,\phi} & \cdots & m_{\omega,\phi} \end{bmatrix} \times \begin{bmatrix} P_\alpha \\ P_\beta \\ \vdots \\ P_\phi \end{bmatrix} \qquad (1)$$

Where the energy flow of the energy hub is defined by input P, to output L, and the coupling matrix defined by m. The set of energy carriers is $\xi$ and the greek indices correspond to the energy carriers $\alpha, \beta, \cdots, \omega \in \xi = \{\text{electricity, heat, ...}\}$. The values in the coupling matrix $m_{\alpha,\beta}$ is the factor of conversion from carrier $P_\alpha$ to carrier $L_\beta$.

This definition can be expanded further to incorporate the converter efficiency's. Each energy input carrier can be split over many converters by a certain dispatch factor $\eta_{i,j}$. Consequently the definition can be changed to:

$$\begin{bmatrix} L_\alpha \\ L_\beta \\ \vdots \\ L_\phi \end{bmatrix} = \begin{bmatrix} \eta_{\alpha,\alpha} & \eta_{\beta,\alpha} & \cdots & \eta_{\omega,\alpha} \\ \eta_{\alpha,\beta} & \eta_{\beta,\beta} & \cdots & \eta_{\omega,\beta} \\ \vdots & \vdots & \ddots & \vdots \\ \eta_{\alpha,\phi} & \eta_{\beta,\phi} & \cdots & \eta_{\omega,\phi} \end{bmatrix} \times \begin{bmatrix} n_{\alpha,\alpha} & n_{\beta,\alpha} & \cdots & n_{\omega,\alpha} \\ n_{\alpha,\beta} & n_{\beta,\beta} & \cdots & n_{\omega,\beta} \\ \vdots & \vdots & \ddots & \vdots \\ n_{\alpha,\phi} & n_{\beta,\phi} & \cdots & n_{\omega,\phi} \end{bmatrix} \times \begin{bmatrix} P_\alpha \\ P_\beta \\ \vdots \\ P_\phi \end{bmatrix} \qquad (2)$$

$$[\mathbf{L}] = [\eta] \times [\mathbf{N}] \times [\mathbf{P}] \qquad (3)$$

where **N** and $\eta$ are the dispatch and efficiency matrices, respectively.

### 2.1 Energy hub model

Since the introduction of the energy hub as a concept, several energy hub models have been proposed in the literature. Figure 2 shows the energy hub model Brahman *et al.*[4] propose. This energy hub is meant to be used on a residential scale, where natural gas, electricity from the net from photo voltaic panels is used to manage demands for hot water, cooling, and electricity. This is done with the use of a combined cooling, heating, and power unit (CCHP). Thermal energy storage (TES) and an electric vehicle (EV) as an electrical energy storage unit are also part of the model.

City-scale energy hub models have also been proposed, like the model proposed by Pazouki *et al.*[8]. A simplified version of this energy hub can be seen in Figure 3. This model is supplied by network energy carriers (electricity and gas) and wind power. The model uses a transformer, AC/AC converter, CHP, and a boiler for converting network energy carriers to demands for the hub. The model has electric and heat storage, and has the potential to sell energy back to the grid as revenue.

Qi *et al.*[9] also propose an energy hub on a city-scale. A simplified model of this can be seen in Figure 4. This model is supplied by energy from the grid and by solar radiation. It uses a concentrating solar panel (CSP) plant, which converts solar radiation to heat and electricity by use of photo-voltaic panels, heat transfer fluid (HTF), a steam generator, a heat recovery unit, and a power unit for electricity and thermal demands. The model also uses a thermal storage system and electric vehicles (EV) as electric storage for varying demands.

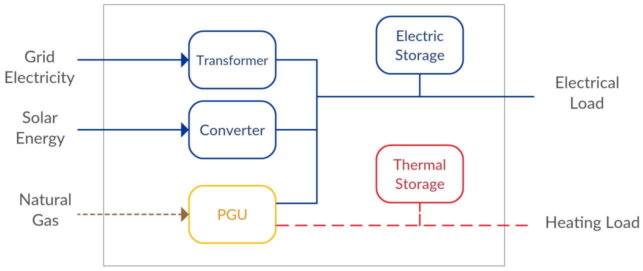


Fig. 2. A simplified model of the proposed energy hub model in the article of Brahman *et al.* [4] for a single household which is supplied by gas and power from the grid together with power from photo-voltaic panels, uses a power generation unit, heat recovery and storage for thermal and electrical demand.

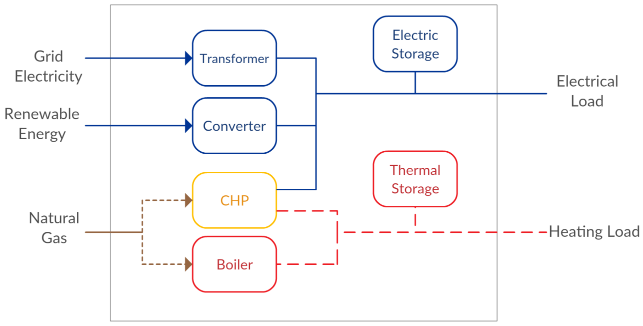


Fig. 3. A simplified model of the proposed energy hub model in the article of Pazouki *et al.* [8] which is supplied by power, gas and water from the grid and user a transformer, converter, CHP, boiler and storage for thermal and electrical demand. It also has the potential to sell energy back to the grid as revenue.

The energy hub model proposed by Alipour *et al.*[1], as seen in Figure 5, is supplied by gas and electricity from the grid and uses a transformer, a CHP and a furnace for electricity and heat demands. It also has power and heat storage for varying demands. This model is proposed as a multi-source multi-product system where systems can be added depending on the context. Alipour *et al.* also performed three studies with this hub to illustrate the effectiveness of the model, both for small and large scale infrastructures.

### 2.2 Comparison of models

To compare the different proposed energy hub models, the components of the models are put into the following four categories: input, conversion, storage, and output.

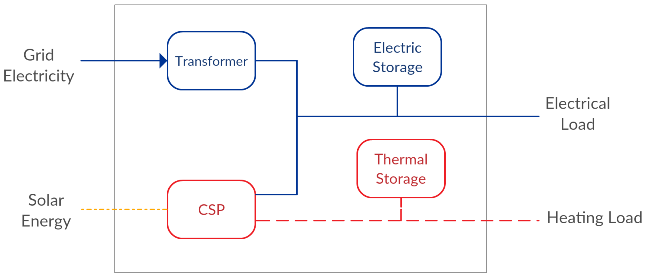


Fig. 4. A simplified model of the proposed energy hub model in the article of Qi *et al.* [9] which is supplied by power from the grid and solar radiation and uses a CSP plant, power unit, heat recovery unit and storage for electrical and thermal demands.

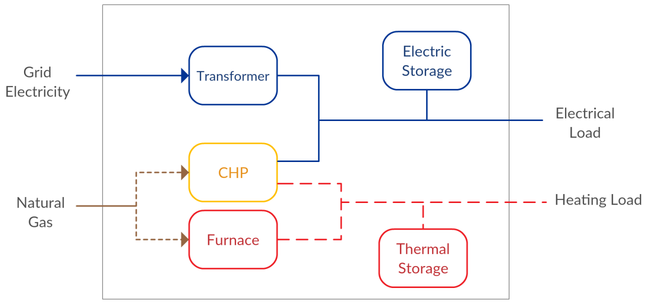


Fig. 5. A simplified model of the proposed energy hub model in the article of Alipour *et al.* [1] which is supplied by power and gas from the grid and uses a transformer, CHP, furnace and storage for electrical and thermal demands.

### 2.2.1 Input

One variable all the energy hubs are supplied by, is electricity from the grid. This makes sense, since most of the countries in the world have some form of a nation-wide power grid. Some of the proposed models also can make use of solar and wind power, which cuts down costs on purchasing power from the grid and lowers CO2 emissions. Note that solar and wind power is always used in conjunction with power from the grid, since renewable sources are too unpredictable to rely solely on it.

Next to electrical power from the net, most of the proposed models also use natural gas from the grid as input. The natural gas converts into power and heat, e.g. by use of a CHP. But this conversion also releases CO2. One proposed model solves this by using a CSP to convert solar radiation into power and heat.

### 2.2.2 Conversion

When gas is included among the energy input of the model, a conversion element is necessary to create the required heat and power. This conversion element converts one form of energy into more or more other forms of energy. A CHP, PGU, boiler, and furnace all convert natural gas into electrical power and heat. Solar radiation can also be converted into electrical power and heat, with the use of a CSP plant. The differences in the methods used for conversion vary according to what the heat is used for, e.g. a boiler is used to heat water, while a furnace is used to heat air.

### 2.2.3 Storage

Electrical and thermal storage are common elements in all the proposed models. Barmayoon *et al.*[2] show that an energy hub benefits from energy storage systems due to energy cost saving and emission reduction. Electrical storage comes in various forms, either battery packs or the use of EV's using the vehicle-to-grid (V2G) function. Note that EV's are not permanently linked to the energy hub, making it an uncertainty for the model to deal with. Heat storage is done with a TES, TSS, or with a hot water storage solution. A hot water storage solution consists of a water tank, used for storing the water, and a conversion unit which transfers the heat into the water. The differences in heat storage is determined by the type of heat demand, e.g. space heating versus sanitary hot water preparation.

### 2.2.4 Output

All the proposed models have power and heat as an output and some of the models also have the potential to return power back to the grid as revenue.

## 3 Optimization problem

The goal of an energy hub is to predict the behaviour of the components of the system in order to determine the optimal electrical and thermal loads to minimize the operational costs or prices. This is a difficult problem since each component within the energy hub has a different impact on the energy streams. Also the inherent uncertainties in the predictions of future thermal and electrical load propagate quick throughout the network. To simplify this modeling problem, optimization methods are used to find approximate solutions.

### 3.1 Objective functions

The optimization problem is solved using an objective function that tries to minimize a specific parameter to achieve the required goals in an efficient way. The optimal operation model can be formulated using multiple types of objective functions. The energy hub can be optimized from an economic perspective, by using the objective of minimizing the costs associated with the power generation, conversion, and storage. Selling and buying energy, e.g. electricity from the network, can also be put in this objective function. Another group of objective functions are based on environmental impacts. An example of such an objective function could be the reduction of $CO_2$, $NO_x$, and $SO_x$ emissions. Combinations of objective functions can be formulated using multi-objective (MO) optimization. This optimization tries to find the best solutions by taking multiple objective functions into account, which often have opposing behaviour; that is, an increase in one objective results in a decrease of another [4].

### 3.1.1 Economic objective functions

Economic objective functions used in reference articles are summarized below as a list of dependent variables in the optimization problem. These variables are minimized on the product of their operational costs times the power that they supply / sell.

| Dependent variables | References |
|---|---|
| Network Power Exchange | [9] [4] [8] [1] |
| Concentrating Solar Panel | [9] |
| Combined Cooling, Heating, and Power Exchange | [4] |
| Electrical and Thermal Storage Power | [8] |
| Combined Heat and Power Exchange | [8] |
| Energy Generation System startups/shutdowns | [1] |

Table 1. Dependent cost variables in economic objective variables in reference articles.

### 3.1.2 Environmental objective functions

Environmental objective functions used in reference articles only involve gas emissions from network electricity and/or gas consumption in their optimization. These emissions can be translated into costs by using certain assumptions.

| Dependent variables | References |
|---|---|
| $CO_2$, $NO_x$, and $SO_x$ emissions | [9] [4] [8] |

Table 2. Dependent cost variables in environmental objective variables in reference articles.

### 3.2 Optimization techniques

This optimization generally results in a trade-off between accuracy and complexity. Optimization problems can be categorized into: mathematical optimization, meta-heuristic searches, and heuristic methods. Mathematical optimization techniques systematically choose input values such that an optimal schedule is obtained [3]:

- *Linear programming* problems are the simplest, where objective and constraints are affine functions. It can be solved in polynomial time, but are less accurate.
- *Quadratic programming* problems are a bit more complex, because of the quadratic objective function. If the objective function is positive definite, it can be solved in polynomial time, else it is an NP-hard problem.

- *Convex programming* problems have a convex objective function. This form of optimization is more complex, but is not guaranteed to converge if a solution exists.
- *Dynamic programming* problems also use convex objective functions, but break complex problems into smaller sub-problems. This reduces the complexity of the calculations and therefore improving speed.
- *Mixed integer linear programming* problems include discrete variables, allowing discontinuities in modeling. Although this is non-linear, it is still an NP-hard problem.
- *Mixed integer non-linear programming* problems also include non-linearity and are difficult to solve and do not guarantee to converge if a solution exists.

Mathematical optimization techniques can be computationally expensive, and therefore meta-heuristic and heuristic methods can be used to simplify the problem. Meta-heuristic methods search over a large set of possible solutions by semi-randomly choosing their path until they converge to an approximate solution. Heuristic methods are knowledge based techniques that introduce rules to approximate the optimization problem. This paper focuses on the mathematical optimization techniques introduced in the models discussed in the last section. Mainly these optimization techniques include MILP and MINLP.

## 4 Energy hub challenges

In this section the problems that arise when combining several types of energy storage, production and / or conversion systems in context to energy hubs are discussed. This section acts as an summary of issues related to the energy hub models presented in section (section 2).

### 4.1 Varying energy hub scale

The energy hubs that have been proposed within the literature vary greatly in scale. One hub focuses on a residential level [4], another focuses on an area of around a thousand households [8] and yet another focuses on a residential quarter of around 3000 households [9]. These differences in scale also translate to a difference in the models for the energy hubs. In order to provide a reliable system all these components have to combined into a coherent and flexible energy hub. Also the optimization problem can be dependent on the scale of the energy hub.

### 4.2 Diversity of household devices

In a modern household people own a multitude of electrical devices that require electricity on different intervals across the day or year. The diversity within the ownership rates of each device varies greatly, as shown by the article of Beaudin *et al.* [3]. This varying electricity demand on a house scale influences the overall character of the loads that impact the demand response program. Therefore it is important the the architecture of the energy hub model is flexible and can cope with this rapid change. Particularly, space heaters, water heaters, air conditioners, refrigerators, electrical vehicles, washing machines, dishwashers, clothes dryers and ovens have the greatest contribution to residential load [3].

### 4.3 Uncertainty in future energy demand

In addition to the varying scale and diversity in devices, uncertainties also play an important role in determining the future energy demand. These uncertainties include energy consumption behaviour, seasonal period, weather conditions, whether solar and/or wind generation is included in the energy hub model and occupancy of a residential area. The energy demand is dependent on the forecasting methods and is therefore highly influenced by the these uncertainties.

### 4.4 Diversity in energy storage, conversion, and generation techniques

As already mentioned in the introduction, there is currently a vast amount of options available for energy storage, conversion, and generation. Concentrating solar power (CSP) units for converting solar radiation to heat and electricity, the use of electrical vehicles (EV) to temporarily charge or discharge the batteries for storage and power generation with the use of combined heat and power (CHP) technologies. Energy hubs should be able to model all types and also every combination of types. This poses difficulties on the accuracy and reliability of the predictions of the energy hub models and needs to be taken into account.

### 4.5 Infrastructural challenges

For the energy hub to function properly it is important that it has access to all the required information, such as weather data, electricity prices, energy demand, etc. Therefore it is essential that the communication between the energy demand components and the energy hub is done in a smart and reliable way. All of these components have to be connected to the internet and function reliably and secure within the network. The energy hub should be able to remain functional even when one of these components is not able to communicate.

## 5 Comparison of optimization techniques

In the analyzed papers two types of mathematical optimization methods have been used: MILP and MINLP. Whenever non-linear behaviour appears within the optimization problem, MINLP can be directly used to solve the problem or it can be linearized and converted into a MILP problem [5]. Whenever a problem is linearized it may not result into the global minimum. Another limitation of linear mathematical optimization techniques is the system must already have reached steady-state, because these methods cannot reflect the dynamic behaviour of the system. In Table 3 an overview is given of the applied type of method.

| Optimization method | References |
|---|---|
| MILP | [8], [4], [9] |
| MINLP | [1] |

Table 3. Mathematical optimization methods applied in reference articles.

Research by Beaudin *et al.* shows that most of the current research is performed using the MILP optimization method [3]. Previous research has shown that experiments with MILP optimization applied shows accurate, reliable and fast results for small to medium sized problems [5]. The field of research on MINLP is also not yet matured enough, since it is more complex and therefore more difficult made reliable [5]. On the other hand MINLP has a main advantage over MILP that it better optimizes large scale systems in which many discrete alternatives exist [5].

This notion of MINLP capable of optimizing large-scale systems is also stated in the conclusion of the article of Alipour *et al.* [1] The rest of the articles implementing the MILP optimization method are all applied for residential scale systems (few thousand houses).

| Method | Reliability | Scalability | Accuracy | Complexity |
|---|---|---|---|---|
| MILP | + | + | + | - |
| MINLP | - | ++ | + | -- |

Table 4. Comparison of mathematical optimization techniques MILP and MINLP applied in reference articles.

The results from these papers is summarized in Table 4 in context to the reliability, scalability, accuracy and complexity. MILP has been investigated more compared to MINLP and can therefore be applied more reliably. Also MINLP doesn't always converge to a solution, while MILP does. MINLP is also less appropriate for achieving high quality exact results in large-scale systems, because of the non linearity [5]. Both methods show good accuracy in their results, but the trade-off is in complexity. Both optimization methods result in an NP-hard problem.

## 6 Discussion

As shown in section 2 every paper uses their own approach in modeling the energy hub and take various energy production, storage, and conversion components into account. Besides having different models, each reference paper also uses their own objective function in order to find the optimal load distribution. The mathematical optimization techniques used in reference article can be categorized into MILP or MINLP, but comparing the relative impact on the reliability, scalability, accuracy, and their complexity is hard to determine. Therefore the results from the paper from Klanšek *et al.* have been used to compare the two techniques. From this comparison it is deduced that the MILP is more reliable and less complex, but MINLP could have a better potential optimizing large-scale systems in which many discrete alternatives exist. MILP is mostly used in the scientific literature of energy hubs and is therefore the most matured technique for energy hubs [3].

## 7 Conclusion

In this paper we showed that an energy hub can be represented by a multitude of different structures, with various energy storage, production, and conversion techniques. There are also multiple optimization techniques to solve the objective function of the energy hubs. All of these are dependant on the scale of the energy hub and the available resources. These models all solve a specific case. Energy hubs face many challenges, from infrastructural challenges to uncertainty in future energy demand. We also compared the different optimization techniques used, with respect to reliability, scalability, accuracy and complexity, where we found that MILP is more reliable, while MINLP is more scalable.

We also found the different energy hub models proposed in the literature to be difficult to compare, because of the different objective functions and techniques used in each hub. In order to increase the implementability of energy hubs as a concept, we think that future research is needed to compare all the different aspects of an energy hub in regards to the scale and optimization. It would be a leap forwards for energy hubs as a concept, if it was clear which techniques work best in which situations, and on which scale.

## Acknowledgements

The authors wish to thank L. Fiorini for her insight and comments that helped improve this paper.

## References

[1] M. Alipour, K. Zare, and M. Abapour. Minlp probabilistic scheduling model for demand response programs integrated energy hubs. *IEEE Transactions on Industrial Informatics*, 14(1):79–88, Jan 2018.

[2] M. H. Barmayoon, M. Fotuhi-Firuzabad, A. Rajabi-Ghahnavieh, and M. Moeini-Aghtaie. Energy storage in renewable-based residential energy hubs. *IET Generation, Transmission Distribution*, 10(13):3127–3134, 2016.

[3] M. Beaudin and H. Zareipour. Home energy management systems: A review of modelling and complexity. *Renewable and Sustainable Energy Reviews*, 45:318 – 335, 2015.

[4] F. Brahman, M. Honarmand, and S. Jadid. Optimal electrical and thermal energy management of a residential energy hub, integrating demand response and energy storage system. *Energy and Buildings*, 90:65 – 75, 2015.

[5] U. Klanšek. A comparison between milp and minlp approaches to optimal solution of nonlinear discrete transportation problem. *Transport*, 30(2):135–144, 2015.

[6] J. Liu, X. Xie, F. Qin, S. Song, and D. Lv. A case study of ground source direct cooling system integrated with water storage tank system. *Building Simulation*, 9(6):659–668, Dec 2016.

[7] M. Mohammadi, Y. Noorollahi, B. Mohammadi-ivatloo, and H. Yousefi. Energy hub: From a model to a concept A review. *Renewable and Sustainable Energy Reviews*, 80(C):1512–1527, 2017.

[8] S. Pazouki, M.-R. Haghifam, and A. Moser. Uncertainty modeling in optimal operation of energy hub in presence of wind, storage and demand response. *International Journal of Electrical Power & Energy Systems*, 61:335 – 345, 2014.

[9] F. Qi, F. Wen, X. Liu, and M. A. Salam. A residential energy hub model with a concentrating solar power plant and electric vehicles. *Energies*, 10(8), 2017.

[10] J. Rifkin. *The Third Industrial Revolution: How Lateral Power Is Transforming Energy, the Economy, and the World.* St. Martin's Press, 2011.

# Hybrid Energy Systems: Optimal operation and demand response

Alexander Lukjanenkovs, Matilda Wikar

**Abstract**—This paper presents how hybrid renewable energy systems (HRES) can be optimized to respond efficiently to the demand of energy production in the system. Many hybrid energy system consist of renewable energy sources, therefore we have a look at how renewable energy sources can be combined and best utilized to make up a well-functioning HRES. In this paper HRES stands for hybrid renewable energy systems, which means the combination of renewable energy sources, like solar panels and wind turbines, that are integrated with the local power grid or operating stand-alone to provide electricity to one or more households. We present more in detail what a hybrid systems are and how they work and what the current issues regarding HRES's are. The paper also describes different methods available for HRES optimization and we propose our own solution, an iterative process by which one can achieve an optimized HRES.

To illustrate how HRES can be integrated to our everyday lives, we will have a look at how HRES can be integrated into a smart home, to not only to optimize the energy generation, but also to use the energy in the most efficient way by decreasing energy consumption without sacrificing usage of any house appliances.

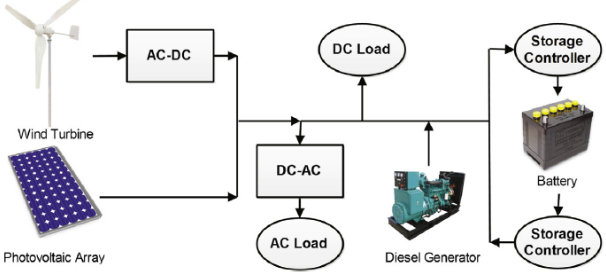
**Index Terms**—HRES, energy optimization, energy consumption, smart home, energy efficiency.

## 1. Introduction

With the global warming and the climate change being a known fact, we face the need for phasing out old energy sources like fossil fuels to make way for cleaner, more sustainable ones. However, today's renewable energy sources (RES) are not yet efficient and stable enough to serve as standalone energy sources for the modern community with its high energy demands. Therefore, hybrid renewable energy systems are a fine solution to this issue. Hybrid energy systems (HES) are the combination of various energy sources, usually with a fossil fueled backup-system. These systems can be preferred due to the stability of the backup, or secondary energy source. The only problem with HESs are that they are not consisting of renewable energy sources, meaning that in the long run they will not be a sustainable option if we want to work against the issues regarding the climate change. Therefore, hybrid renewable energy systems (HRES) would be to prefer over traditional HESs, as the clean energy sources would be utilized all the way through the process of harvesting energy to using it for our needs. In this article we go through various methods available for optimizing a HRESs, because we believe that there will not be one single solution that results in these systems being optimized, but many ways that combined together will achieve this. The factors that have impact on how well a HRES is performing are many, but first we look into how the to find the best design based on the available sources and how the location of these matters in the design process. As we are dealing with RESs, which are sensitive to the conditions surrounding them, location and availability have a great impact on the system, but that is not all that can be done. Optimizing the operation of the HRES and introducing demand response programs (DRP) can have an impact not only on how much energy is produced and demanded at the moment from the system, but can also serve as an economical motivator for the users to plan their activities regarding electricity usage. The economical factor is also to be taken into account when optimizing HRESs, both for the provider of the system as well as for the end customer. Another important question within the technology of RESs is how to ensure a stable production of energy. What the factors are to take into account when optimizing an operating HRESs are described, and methods for this also. This paper also describes our own approach for designing a HRES in multiple steps, taking into account that this is an expensive process. In Section 3 we aim to discuss, besides the economical factor in the design process, also how the system design can be optimized while already being operative. In Section 4 we give a small description of the most pressuring issues before describing our own approach in Section 5. In section 6 we also want to propose that much can be done also in the consumer household by trying to lower the demand for electrical energy. Section 7 discusses the future of HRESs and in Section 8 we give a conclusion of the paper.

## 2. Hybrid Renewable Energy Systems Definition

Fig. 1. HRES schematic.[14]



In this paper, we consider an HRES that consists of multiple components, namely wind turbine, solar panels, diesel generator and batteries.

The characteristics of each system component are the following:

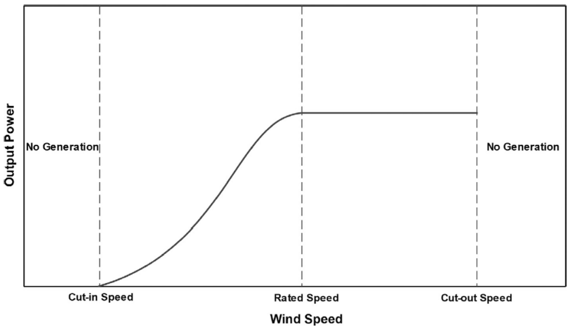
**Wind energy conversion system.** It is a component which produces energy in combination with other energy producing components of the system. Fig. 2 shows a typical curve of the relationship between the output power and the wind velocity. When the wind speed is less than the cut-in speed (normally $3-5$ m/s), no power is generated. Between the cut-in speed and the rated or nominal wind speed (normally $11-16$ m/s), the wind turbine output power is directly proportional to the third power of the wind speed. When the wind speed exceeds the nominal value, the output power needs to be limited to prevent damage to the generator and the corresponding power electronic devices. If the wind speed is higher than the cut-out speed (normally $17-30$ m/s), the system will be stopped to protect its components [5] .

**Solar energy conversion system** Solar energy is transformed to electricity through a system which contains a photovoltaic (PV) modules to convert sunlight into electricity, and an inverter to change the direct current (DC) from the modules into alternating current (AC).

**Backup diesel generators** can supply unmet load requirements that exceed renewable generation to increase the reliability of the power

- *Alexander Lukjanenkovs is a MSc. Computing Science student at University of Groningen. E-mail: a.lukjanenkovs@student.rug.nl.*
- *Matilda Wikar is a MSc. Computing Science exchange student at the University of Groningen. E-mail: m.j.wikar@student.rug.nl.*

Fig. 2. Relationship between wind power output and wind speed.[14]



systems. Besides meeting demand directly, diesel generators can also be used for battery charging.

**Deep-cycle lead-acid (DCLA) batteries** which have a typical efficiency of $85-95\%$ are the most advanced technology to be used as storage in a renewable energy system. Although lead-acid batteries have relatively small volumes of energy compared to other types of batteries, they are advantageous due to their low cost and high reliability [6]. By battery efficiency we describe the coulombic efficiency, which is the ratio of the number of charges that enter the battery during charging compared to the number that can be extracted from the battery during discharging. The losses that reduce coulombic efficiency are primarily due to the loss in charge due to secondary reaction, such as the electrolysis of water or other redox reactions in the battery. In general, the coulombic efficiency may be high, in excess of 95%.

There might be multiple alternative schematics of the HRES, for example, instead of using diesel generator, household can be connected to the local electricity grid or to the local gas grid. But for the sake of simplicity we will discuss and work with one version of HRES, which is defined in the previous sections.

## 3. METHODS FOR OPTIMIZATION

Optimizing can be of a wide spectra when discussing energy systems. Aspects that have to be taken into account when optimizing energy system are not only related to the performance, but can strive to decrease the economical factors in the system by for example lowering the consumer-price. Ideally, optimizing would mean minimizing consumption while securing a stable energy production with as little loss of electrical energy. This again means that there is not one single parameter to optimize within HESs, but many different factors, like for example availability and location, to look into to while optimizing this kind of system.

With HRESs being able to operate in either stand-alone mode or in grid-mode, optimization of both modes are needed. Both of these modes result in different problems, meaning that the optimization methods will be focusing on either stand-alone or grid-integrated HRESs, or how to combine these to result in all-time optimized system.

Since one of the main issues related to wind and solar power generation is their dependency on real-time weather conditions, a relatively easy way of optimizing the energy system is to take action based on the weather forecast. By predicting the solar irradiation and the speed of the wind, the HRES can respond based on the data, making it able to prepare for these renewable energy sources not being ideally accessible. Utilizing this data to provide for day-ahead scheduling to meet the demand of the system is a popular method presented in several studies, e.g., [14] and [17].

There are two aspects within optimizing of HRES - the demand response (DR) and energy efficiency (EE), which both are important for reaching a fully optimized system. By taking into account both demand response and energy efficiency optimization it is possible to optimize the whole chain of the electrical energy in the system. This means having the energy production optimized to be as energy efficient as possible, and at the same time follow through the process by having the demand response optimized, for example by having consumers looking into their behaviour.

### 3.1 Design Optimization Based on Available Sources and Location

As already mentioned, the performance of RESs, such as wind and solar power, are sensitive to the weather circumstances surrounding them. Different climates have different seasons with more or less suitable pre-conditions for example solar panels to function well. This means, that there is no single setup of components within a certain HRES available that would give optimized electrical energy anywhere on the globe, because of varying climate. This issue results in the need of every HRES involving these energy sources to be evaluated based on the pre-conditions for harvesting RESs. Wang [14] proposes an iterative procedure for reaching optimized performance of a HRES based on what components are suitable for the specific system. This process takes into account both the optimized design of the system and the economical aspect of designing a HRES. The procedure is based on the cost of each component for the combined wind and solar-power system as well as how much electrical power the component is capable of producing. For example, the cost of one wind turbine can be as much as the price for ten solar panels. This is to be taken into account when deciding on which energy source is the most suitable for the specific HRES, as also the difference in how much energy is produced by both sources.

The steps are as follows: first, select wind turbines, solar panels and batteries for storage based on their return of investment (ROI) value. The ROI value describes the efficiency of the investment and is calculated by dividing the economical return of the investment by the original cost. After setting on the components, fix the number of wind turbines as they are the more expensive. Combined with the solar panels, wind turbines often give the best performance, so the next step is to increase the number of solar panels to the HRES. This step can then be re-executed if the produced electrical energy does not meet the demands. When the amount of wind turbines and solar panels are fully optimized, the storage need for the system should be re-calculated with regard to the new setup. Based on that calculation, batteries should be added to meet the need of the system definitely. [14]

### 3.2 HOMER Software Tool

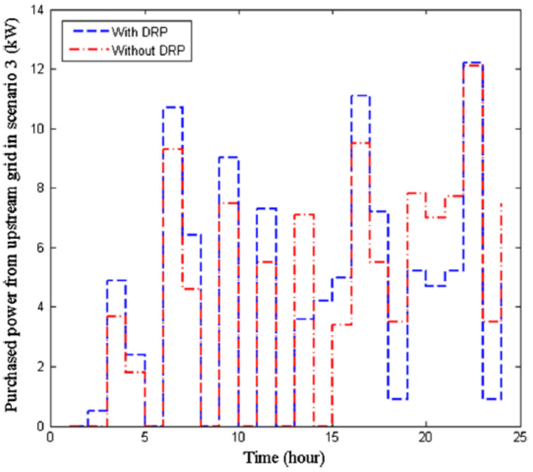
There exists a model of Hybrid Optimization Model of Electric Renewable (HOMER) that have been developed and tested in Canada. [11] HOMER is a software tool for sizing and optimization of the HRES. HOMER acts upon various models of renewable energy source components and by taking into account the available components and the cost of each. For correct analysis with HOMER, it needs apart from those facts also information about the control methods of the system, and how efficient each component is. HOMER is one example of how the system can be optimized and simulated beforehand, to ensure the energy efficiency. An energy efficient base of the system is crucial to have before adding further optimization methods, such as DRP. Having the system optimized already in the design phase is a strong quality, which can then be built upon with the other concepts described earlier. [11]

### 3.3 Operation Optimization by Demand Response Program

There are two main benefits within a demand response program (DRP). The first one is a more stable consumption that also regards the availability of electricity based on the conditions for harvesting RESs. The second one is for the consumer, who can benefit economically by avoiding high consumption while electrical energy is expensive during worse conditions for energy production. With day-ahead scheduling of their demand for electricity, the consumers are asked to plan the activities of the household based on the predictions of electrical energy production available. This data is to be provided by the supervisor of the grid, based on real-time weather data. This can be done by looking into the power consumption of every home appliance and based on that the user can decide to, for example, move the usage of a highly energy consuming home appliance to a day with a cheaper electrical price. The core of this kind of optimization is to find a trade-off between minimizing the costs and the waiting time for the customer [14]

To make it more understandable for the consumers so that they can decide on when and how to adjust their energy consumption, a so-called cost term is introduced. This cost term can then again be utilized to evaluate the loss-of-supply (LOS). LOS describes here how important or necessary an appliance is, by evaluating how unacceptable the loss of the appliance would be. The higher the LOS, the less willing the customer probably is to change the planned energy-consuming activities. This is often the case for necessaries such as air conditioning or heating, as well as refrigerators and other every-day appliances. In table X examples of how much power these appliances consume can be viewed, divided by the likelihood of the consumer moving the usage to a day with lower LOS. The motivational factor for the customer is the reduced price. [14] [8]

Fig. 3. Purchased power from the grid. [8]



Experiments have been conducted comparing two similar hybrid energy systems, one with and one without the Demand Response Program (DRP) introduced.[8] This study shows that the DRP has an impact on the system in multiple ways. For example, it affects the load profile by causing it to be more fluctuating, due to the customers probably knowing better when it is efficient to use the power. The same phenomena can be seen when observing the peaks in the purchased power from the grid, with and without the implementation of DRP, see Figure 3. The same study showed that implementing a DRP resulted in a flatter load curve, when load from expensive peak periods shifted to off-peak periods which are cheaper in terms of cost. The authors [8] state that this, besides being more cost-friendy, has an impact on the lifespan of the components in the system. This is because when they are under a more consistent load, they last longer.

### 3.4 Operational Optimization

With an ideal setup of an HRES, it remains still to optimize the performance during its operation. With relatively unstable energy sources, forecasting of the factors impacting on the energy production, such as wind and solar irradiation, is highly valuable. By forecasting these circumstances for the production of electrical energy, the system can be alarmed if a period with worse conditions is coming up, for example if a rain-period with little sunshine is to be expected, or if the wind velocity will be too high or low for the wind turbines to operate on. Then backup-systems can be activated or the energy generated beforehand can be stored in backup-batteries.

Taking precautionary measures may however, not always be enough. Backup-systems and storing energy are both good ways of acting, but it should be complemented with real-time optimization to ensure the best performance of the system. Again, real-time forecasting of the weather circumstances might be used on such small intervals as 15 minutes. The weather data can be achieved by statistics or measurements of the environment, or artificial intelligence such as neural networks. This data can be used for system management with a short response time, making it efficient.[14]

Table 1. Rigid and flexible home appliances with their power consumption in watt[14]

| Type | Electricity appliance | Power/watt |
|---|---|---|
| Rigid Demand | Air conditioner | 3000 |
| | Heating | 4000 |
| | Ceiling fan | 100 |
| | Refrigerator | 600 |
| | Lights | 20 |
| | Television | 200 |
| | PC | 120 |
| | Monitor | 150 |
| | Laptop | 70 |
| | Coffee maker | 600 |
| | Microwave oven | 1300 |
| Flexible Demand | Dishwasher | 1200 |
| | Vacuum cleaner | 220 |
| | Washing machine | 300 |
| | Clothes dryer | 2200 |

Another approach to optimize operational HRESs would be so-called receding horizon optimization (RHO). Receding horizon optimization derives from a well-known model within system process engineering. RHO is partly based on economical models but can be applied on HRESs. This is done by trying to minimize the costs within a certain time-horizon evaluating the cost of each HRES-component individually.[14]

Simplified, the cost function in RHO is acquired by summarizing the operational costs with the environmental costs. These factors are then multiplied with the amount of electrical energy generated from each of the sub-components of the energy system. Based on these calculations it is possible to determine which energy sources that are most efficient to utilize when producing electricity.[14]

## 4. Problems

Due to a decrease in traditional fossil fuel resource amount and increase in the worldwide energy demand, alternatives of energy technologies are getting a significant attention. By using renewable energy technologies, evident environmental advantages can be observed and as well the dependency on the fossil fuel resources and carbon emissions to the atmosphere are decreasing. [12].

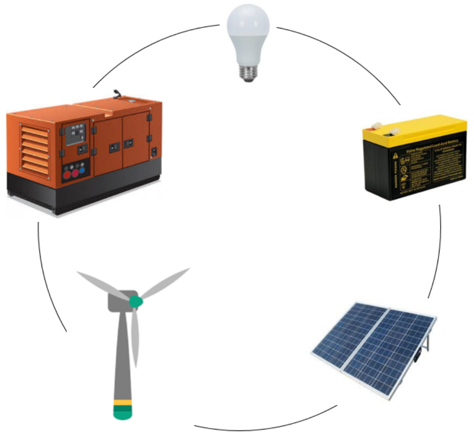
Several studies have examined modeling, control and optimization of hybrid energy systems, starting from the design to operation[10, 13, 3, 9]. Article [2] by Bajpai and Dash (2012) thoroughly reviewed HRES for power generation in stand-alone applications. The article stated that the major concern for renewable energy penetration into the grid are the high manufacturing and maintenance costs of the wind turbines and photovoltoic (PV) panels.

## 5. Our proposed approach towards the solution

One part of the proposed solution is to decrease the power consumption of the household without sacrificing the number of used home appliances and user's comfort. There exist multiple simple solutions to achieve that. First, one solution is to start to use efficient electrical appliances, for example, led lights instead of regular light bulbs, efficient low energy usage fridge, etc. Another solution, although it may be more complex, is introducing controllers for the appliances. This solution might include simple motion detection/timer controller for the household lights, or it also can include more complex controllers, which can analyze and predict environmental variables. For example, a controller for the household's air conditioning system. This controller will analyze and predict which room will be used and what time. In such way only occupied and used rooms will be cooled, meanwhile other rooms won't be cooled. By implementing this controller household will save energy on the cooling while at the same time comfort of the house users are not affected.

We propose an iterative solution based on multiple steps (Figure 5), which can be fulfilled without following all steps in the instructions. Namely, with our solution the system can start operating from any number steps completed (in the right order). It is not a requirement to finish all steps to optimize the system of the house , but it is still a goal, which needs to be reached with the time, if the consumer wants a full capacity of optimization for the system.

Fig. 4. Our proposed model for increasing energy efficiency and moving towards a stand-alone HRES.



It is understandable that not everyone can implement full scale HRES system, for multiple reasons. For instance, it is too complex or too expensive, or both together. That is why, we designed a multiple step iterative solution, in which any amount of steps in the order can be done to start optimizing the current home system. By iterative we mean that at first the user can install a smaller version of HRES for his house than actually required. The first version of the system can be as big as the user can afford and build at the time. And later expand the system by following the steps in circle, like illustrated in Figure 4, to scale up the HRES for the needs of the house. Implementing the system with multiple small iterations until it satisfies the power need of the house will make this approach more affordable to the average household. Also, the smaller version of HRES will already start saving money from the start, and this saved money can be further invested towards HRES expanding.

Fig. 5. Our proposed model enlisted in five steps.

Step 1: Small scale optimization of the household by using smart sensors/controllers and without adding specific HRES components.

Step 2: Investing in batteries to store energy.

Step 3: Adding solar panels to generate your own electricity.

Step 4: Adding wind turbine(s) to increase energy production capabilities.

Step 5: Adding a backup-generator for the redundancy to improve systems reliability.

Each step of the Figure 5 is discussed in the next following paragraphs.

First step focuses on already existing appliances and it aims to add smart sensors and controllers to these household's appliances. These sensors and controllers are affordable and will allow to achieve some energy savings. The amount of energy saved will depend on the number of appliances improved with the controllers and the complexity of these integrated controllers. More complex controllers has a potential to save more energy than a simple controller. By a complex controller we mean a controller, which analyzes the historical data and input from the sensors to make according predictions or decisions. Later, these predictions and decisions can be applied to specific appliance, which will decrease a power consumption of this appliance while not affecting the experience and comfort of the user. One example of such controller was discussed in the first paragraph of this section, example about air conditioning system. This step is the easiest and most affordable one, meanwhile other steps focus on adding new components for the HRES which are more expensive. Importance of this step is that by reducing the power consumption of your household will make your house require to install a smaller scale HRES than before the power savings. And if a smaller version of HRES will suffice your household needs then it will be noticeably cheaper to implement this system, as the total cost of HRES components is not cheap.

Second step is to get a battery which satisfies the energy needs of your house. If it is too expensive to get all set of the batteries needed for your house from the start, then it is possible to start by implementing as many batteries as the budget allows and later expand your system with more batteries until the total battery size matches the household electricity demand size. First you can setup this acquired batteries in certain way that these batteries will charge during the night time, when the electricity price is the cheapest, and use this saved electricity during the day time, when the electricity price is at the highest level. It is suggested to do, because at beginning your system will consist only from the batteries without any HRES energy producing components. So this will allow the household to start saving some money in your electricity bills from the start. This certain setup is suggested, because during daytime average household uses more energy than during a night time. Therefore, during daytime when more electricity is used, the house will use the cheapest night time priced electricity, by that saving some money.

Third step consists of adding some solar panels to the system. It is possible to start with one solar panel and with the time expand the system by adding more solar panels to match the house energy demand. The average consumer can use the saved money from the previous steps to invest it in this step implementation. It is important that this step is implemented after the second step, which is about batteries, that there is where to store the produced energy, and also that this produced energy is not wasted.

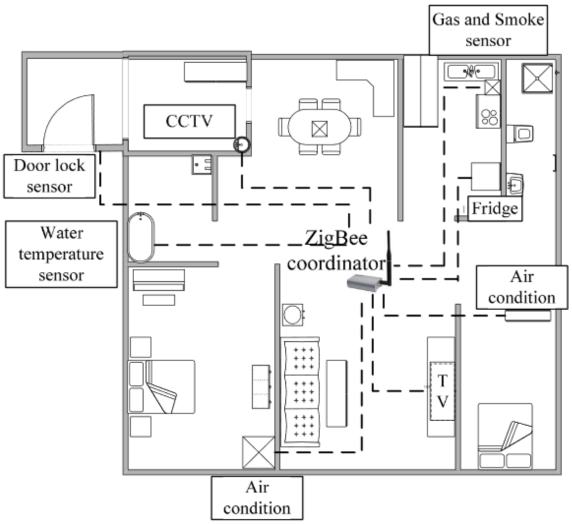
Fourth step (recommended step), the user can also install wind turbines to combine it with the solar panels. This is a recommended step, because we understand that not everyone will have an option to install the wind turbine, as wind turbine will require extra free space of land and also wind turbines have strict safety and noise regulations from the government, for instance, wind turbine cannot locate closer than certain distance from the residential buildings. Other reasons are that usually solar panels suffice the average household needs of energy production and also there are no steps of the solution which would require or depend from this step. But if there is such an option to implement this step, then it is very handy to do, because solar panels are effective only during the peak times of sunny days, and wind turbine(s) can help to compensate for the luck of sunny days. In addition, they also can work at night, whereas solar panels cannot. Therefore, this addition can help to ensure stability of energy supply to the household and independence from the grid.

Fifth optional step could be adding a diesel generator as a backup. If something in the system breaks down or something else happens, for instance, energy production components cannot match the demand (which indicates that one of the previous steps components need to be expanded more), and resulting that this system is nonfunctional, then this step will ensure that the household will still have the power to use and operate. This component is planned to be used for the emergency situations only, and not for regular everyday use.

## 6. An Example of a Smart Home

There are various proposals of smart home architectures, of various sizes and with different approaches. Here we present a simple, doable approach on a smart home that could be combined with a HRES to reach both optimized energy production as well as optimal conditions for the energy usage of the home. The ZigBee standard have proven itself to be useful when it comes to automated homes. ZigBee is energy efficient operating on a low energy level, and it can efficiently be translated to communicate over the WiFi-protocol, but can also operate on its own. [16]

Fig. 6. Layout of a house with ZigBee-connected components [16]



The ZigBee-based system could include a various amount of different components. For example, the control system setup could include motion sensors in every room to detect whether the lights are actually needed. The system could also integrate air condition and heating to optimize their performance based on whether they are needed or not. See Figure 6 for layout of a home integrated with Zigbee. [16]

As proposed in our model for adapting a HRES in segments based on the possibilities, this is to illustrate the optional but much preferred step of including controllers to further increase the energy efficiency in the household.

## 7. Future of HRES

Hybrid renewable energy systems could fill a gap on the market for available, relatively cheap solutions for areas not connected to the grid. In the world there is still over one billion people without electricity, and stand-alone energy systems of this kind are a strong option when it comes to decreasing the amount of people living of the grid [1].

Local HRESs can compete with the traditional energy generation technologies, like, hydroelectric or nuclear methods of producing electricity, by their relatively simple setup and the winning factor of less power loss during transmission over a larger grid. However, there is still work to be done on improving the stability of HRESs, and to overcome the different environments around the world. With a climate this varying, the same solutions cannot be applied everywhere and every location on the globe provides its own difficulties regarding the conditions to utilize renewable energy sources[4].

## 8. Conclusion

Optimizing HRESs as seen in this paper, not only consisting of one straightforward task, but there are many methods that combined together can result in an optimized HRES. However, we see a great potential regarding the technologies connected to renewable energy sources as well as for the different parts making up for the optimization of HRESs. In combination with smarter homes, that are more optimized to have lower energy consumption and thereby less demand for electrical energy, we are convinced that it is possible to make energy systems completely out of renewable energy sources without having to adjust all that much of the quality of the everyday life.

Starting from the beginning in the design phase, the HRES can be optimized to operate with ideal settings for the HRES in question. By using the iterative method proposed [14] or software tools like HOMER [11] designed to be utilized in the process of setting up a HRES, optimization is taken into account immediately. When designing with care, also the economical factor is regarded. HRESs are not cheap, with the price of one single wind turbine being around 16,000€.[14]

DRPs showed to have a great impact on HRESs and plays an important role when optimizing these energy systems, as DRPs not only eases the demand for energy, but also have another positive effect on the system prolonging its lifespan.[8]

With our proposed model we wanted to take into account both the economical aspect of designing a HRESs and follow the iterative design model proposed by researchers on the topic of operation optimization.[14] We believe that the model would work and that it is a relatively easy way of accomplishing the setup of a HRES. The smart home presented could maybe not be regarded as part of the HRES in particular, but by taking it into account we hope to illustrate the bridge from HRES technology to smart-home technologies, and how they together in a bigger perspective, both results in less electrical energy being consumed.

New technologies are always developed and we hope to see in the future more energy efficient main components that can be used in the design of HRESs, as well as even more and better optimization methods and tools. Hopefully, attitudes towards cleaner energy alternatives will continue growing positive, and with the legislation promoting renewable energy will motivate investments in HRESs.

## References

[1] International Energy Agency. World energy outlook. 2017.

[2] Dash V Bajpai P. Hybrid renewable energy systems for power generation in stand-alone applications: a review. *Renew Sustain Energy Rev 2012*, 2012.

[3] Dufo-Lpez R Bernal-Agustn JL. Simulation and optimization of stand-alone hybrid renewable energy systems. 2009.

[4] Gil-Yong Lee Young-Man Cho Sung-Hoon Ahn Binayak Bhandari, Kyung-Tae Lee. Optimization of hybrid renewable energy power systems: A review. 2015.

[5] Wang C. Modeling and control of hybrid wind/photovoltaic/fuel cell distributed generation systems. 2006.

[6] Berndt D. Maintenance-free batteries: lead-acid, nickel/cadmium, nickel/hydride: a handbook of battery technology. 1993.

[7] Venkataramanan G-Gerez V. Kellogg W, Nehrir M. Generation unit sizing and cost analysis for stand-alone wind, photovoltaic, and hybrid wind/pv systems. 1998.

[8] Kazem Zare Majid Majidi, Sayyad Nojavan. Optimal stochastic short-term thermal and electrical operation of fuel cell/photovoltaic/battery/grid hybrid energy system in the presence of demand response program. 2017.

[9] Tiano FA Marano V, Rizzo G. Application of dynamic programming to the optimal management of a hybrid power plant with wind turbines, photovoltaic panels and compressed air energy storage. 2012.

[10] Celik N-Lee S Son YJ Head L Mazhari E, Zhao J. Hybrid simulation and optimization-based design and operation of integrated photovoltaic generation, storage units, and grid. 2011.

[11] M.T. Iqbal M.J. Khan. Pre-feasibility study of stand-alone hybrid energy systems for applications in newfoundland. 2005.

[12] Chmielewski DJ Soroush M. Process systems opportunities in power generation, storage and distribution. 2013.

[13] Nehrir MH Wang C. Power management of a stand-alone wind/photovoltaic/fuel cell energy system. 2008.

[14] Nael H. El-Farra Xiaonan Wang, Ahmet Palazoglu. Operational optimization and demand response of hybrid renewable energy system. 2015.

[15] Ilic MD Xie L. Model predictive economic/environmental dispatch of power systems with intermittent resources. 2009.

[16] Jianbo Liu Jianping Chai Yepeng Ni, Fang Miao. Implementation of wireless gateway for smart home. 2013.

[17] Yuanzhang Sun Congying Wei Jing Wang Deping Ke-Xiong Li Jun Yang Xiaotao Peng Bowen Tang Yibo Jiang, Jian Xu. Day-ahead stochastic economic dispatch of wind integrated power system considering demand response of residential hybrid energy system. 2017.

[18] Toshiya Saito Kota Suzuki Masaaki Hirayama Akio Mitsui-asao Yonemura Hideki Iba Ryoji Kanno Yuki Kato, Satoshi Hori. High-power all-solid-state batteries using sulfide superionic conductors. 2016.

# A Review of Session Type Systems

Dan Chirtoaca Thijs Klooster

**Abstract**— Process calculi are small programming languages designed for describing communications and synchronizations between collections of independent processes. Session types represent a formalism used to model structured communications. They allow to explicitly specify correct communication behaviour. A handful of process calculi have been developed as well as several session type systems for these process calculi. Each of these typing systems is focused on enabling specific features, thus the differences between them determine their expressive power and usability.
Our research investigates the expressiveness of several session type systems by analyzing the syntactic as well as semantic differences between them. DILL, CLL, CP and GV are the four session type systems we review. We discuss the identified characteristics and reason about their impact on concrete usage scenarios.

**Index Terms**— Process calculus, pi calculus, linear logic, intuitionistic logic, session types, proof carrying code, concurrent systems.

## 1 Introduction

Advances in both software and hardware allow for the creation of bigger and more complex pieces of software annually. These programs might consist of millions of lines of code, maybe even allowing for a form of intelligence up until a certain degree. Such programs may also be concurrent ones, meaning that several of their components run in parallel, while communicating with one another. These types of systems are for example implemented by medical software, on-board vehicle software or military software. The application and influence of such systems determines the degree of correctness those pieces of software need. More specifically, since we are talking about concurrent systems, the correctness of the communication between components of these systems is crucial. The application of a specific system determines the main goal for which that system is used. The influence of a system is the degree in which the system can affect outside factors. Critical systems, safety-critical, business-critical or otherwise, should not contain bugs that prevent them from functioning properly. Malfunctioning systems could negatively affect real-world factors like the health of certain people, which is obviously undesirable. Unfortunately, determining whether communication is done correctly is very difficult, especially with large, complex, and concurrent pieces of software.

The comparison of different fields of study can result into useful insights, through the analogy between those fields. The field of computation has been linked with the field of logic by groups of people interested in software communication correctness. A new principle is introduced, named Propositions as Types, which is also known as the Curry-Howard Correspondence or Formulae as Types [12]. Propositions as Types shows a correspondence between some programming language and some form of logic. There are three main statements within this notion, namely that we can view

**propositions** as **types**,
**proofs** as **programs**, and
**normalization of proofs** as **evaluation of programs**.

This means that each proposition in the logic can be read as a corresponding type in the programming language, as well as the other way around. The same goes for programs and proofs in the logic, each program has a corresponding proof and vice versa. Also, every possible way to simplify a proof has a corresponding way of evaluating a program. This means that there is an isomorphism between these two sides of Propositions as Types, since they are analogous to one another.

Propositions as Types applies to many types of logic. It is also an inspiration for $\lambda$-calculus, programming languages like Haskell and Scala, and automated proof assistants. The notion of Propositions as Types explains aspects like functions, variants, and session types [12].

As previously stated, correct communication in a distributed concurrent system is essential. In order to be able to make concrete claims about the correctness in communication, Propositions as Types is used as a basis to build upon. Session types are a result of this. Session types are a type discipline attributable to making specifications about the communication behaviour of programs. Such a specification is a concise description of the continuous interactions between components in a concurrent system. It allows for programs to be checked, in order to determine whether they comply with the communication protocols. Session types can be interpreted as ”types for protocols”, which essentially builds on the concept of data types. A protocol is a system of rules which two or more entities need to follow in order to achieve the desired communication. Thus, the idea behind session types is defining a formal way to describe communication protocols as a type.

Within the computation field of study, this paper focuses on session type systems, as well as the differences between a number of them. The research question therefore is

*‘What are session type systems and how useful could some of them be?’*

To answer this question, several session type systems are analyzed. These include: DILL, CLL, CP and GV. We explore how one typing system improves on the other, and possibly trades in on other aspects. Additionally, some use cases of these session type systems are discussed.

## 2 Session Types

A session defines a series of interactions which together serve as a unit of conversation. Session types are a type discipline used in communicating processes. They describe sequences of actions that determine the types of messages being sent and received. The communication of the interacting processes is performed along channels that allow for information exchange. Structured sequentiality of operations in session types is what makes them suitable for modeling structured communication-based programming [5].

Introduced as a theoretical foundation for modeling protocols, session types guarantee properties such as privacy and communication safety as well as session fidelity and progress. In this context, a protocol represents a system of rules that allows two or more entities of a communications system to transmit and receive data.

- *Dan Chirtoaca is with the University of Groningen, E-mail: d.chirtoaca@student.rug.nl, ID: s2785307.*
- *Thijs Klooster is with the University of Groningen, E-mail: t.klooster.1@student.rug.nl, ID: s2652781.*

Privacy is ensured by the fact that the session channels are known only to the processes that participate in the session and use them. The session channels have a known structure. This guaranteed by the aspect of session fidelity. In turn, this translates to communication that follows a predetermined order and communication that consists of correct message types. The actions that take place within a session, occur in dual pairs: when a process sends something, the other is responsible for receiving and vice versa. When a selection is offered, the other process makes a choice. The communication is ended when the session terminates. Compatibility between communicating processes is ensured by the aspect of duality [7]. Duality refers to interactions that are complementary. This leads to communication safety, as there can be no communication error. The channels in a conversation are linear, which means that they are guaranteed to occur in exactly one interaction (session) at a time [10]. The structured nature of session types together with the property of linearity of session channels ensure progress. Progress is property that assures eventual completion of the operations.

In order to understand what session types look like, we present a small example that also aims to elaborate on the property of duality. Consider two interacting processes $A$ and $B$. The communication that they establish consists of $A$ sending a string to $B$. Let this be the product id in a catalog, followed by a number representing the amount. Then, process $B$ answers with a boolean value, if the inquired product is available in stock in the requested quantity, and ends the session. This communication would then have the following type:

On the channel endpoint connected to $A$:

$$!String.!Integer.?Boolean.end$$

On the channel endpoint connected to $B$:

$$?String.?Integer.!Boolean.end$$

The notation identifies a sequence of actions by their type. Actions are chained with (.), (?) determines the ability to receive a value of the provided type, and (!) corresponds to sending a value of that type. What we notice is that indeed the connection is dual. The session on either end has complementary actions on the other side - a send is bound to a receive and vice versa - therefore, the conversation between the processes occurs without any mismatch.

Different categories of session types have been created. Up to this point we have focused our attention on simple, binary session types which correspond to binary protocols that involve two partners.

Dependent session types are session types that, besides allowing for describing input and output behaviour of communicating processes, allow for describing the data that is being exchanged as well. The need to know what data is transferred follows from the idea that receiving data that only corresponds to the syntax is not sufficient. The semantics of the data must also be correct. This would allow to eliminate redundant checks and computations. Following Pfenning, Caires and Toninho [8] we describe dependent session types in Section 5.1.

Multiparty protocols may involve more than two parties, hence, binary communication would not be sufficient. Multiparty session types have been created in an attempt to solve this issue. They are used to describe interactions involving multiple concurrent peers. It naturally follows that reasoning about the properties of such systems becomes increasingly harder. Nevertheless, a formal relation between two existing theories of binary and multiparty session types has been presented in the paper by Caires *et al.* [1]. We take a further look into multiparty session types in Section 5.2.

## 3 Methodology

In order to answer the research question presented before, we take four session type systems and investigate them to obtain insights about what they are and their usefulness. This investigation is done by finding papers that cover these session type systems, and analyzing those. First, the papers that introduced the four session type systems in question are consulted. This is used as a starting point for finding additional papers, by taking a look at the referenced papers that might be useful for this research as well. We also look at papers that present session types in a more general sense, as well as papers that compare several of the systems we are interested in to one another. This makes for the basis of the information source of the research conducted here.

For each of the four analyzed session type systems, we first describe the system in a general sense. We present the origin of the system, as well as the syntax that comes with the system. Some systems may be more useful for certain aspects of describing software communication correctness than others. Some systems may lack factors, which makes them less useful for other applications. These aspects are covered in a discussion based on the information extracted from the papers that we analyze. Additionally, some examples of models of real scenarios in the session type systems are shown and explained. These examples are based on the ones presented in the referenced papers. They are adapted and simplified, for better understandability and for presenting the syntax more clearly.

Some of the identified papers present extensions of session types, which are covered in this paper as well. All of the information presented for these extensions is again extracted from these papers. Later on, the usability and expressiveness aspects for the presented session type systems are also summarized. These aspects are based partially on our personal interpretations that we got from analyzing the aforementioned papers, and for the other part, on the opinions of the authors of the referenced papers.

## 4 Session Type Systems

Several session type systems have been and are still being developed. In this section, we present four session type frameworks, introduced in Section 1. DILL and CLL, which are compared in the paper of Caires, Pfenning and Toninho [3], are type systems for the $\pi$-calculus. This is a class of process calculus, which allows for channel names to be communicated over the channels themselves. Using this $\pi$-calculus, describing dynamic communication is made possible. As for the other two type systems, CP and GV are presented in the paper of Wadler [11]. CP is inspired by DILL, while GV is a multithreaded functional language with session types, employing a more verbose syntax when compared to the other three.

The characteristics of these systems are further described below. In order to get a better understanding of the foundation upon which some of these systems are built, the formal definition of a process in $\pi$-calculus is given in Figure 1. This definition is used in some of the examples given when discussing the four session type systems later on.

$$\begin{array}{rlr}
P ::= & \mathbf{0} & \text{inaction} \\
\mid & P \mid Q & \text{parallel composition} \\
\mid & (\boldsymbol{\nu} y)P & \text{name restriction} \\
\mid & x\langle y\rangle.P & \text{output} \\
\mid & x(y).P & \text{input} \\
\mid & !x(y).P & \text{replicated / shared input} \\
\mid & x.inl;P & \text{left choice} \\
\mid & x.inr;P & \text{right choice} \\
\mid & x.case(P,Q) & \text{case offer}
\end{array}$$

Fig. 1. Formal definition of a process in synchronous $\pi$-calculus as introduced in the paper of Sangiorgi and Walker [9], extended with choice as presented in the paper of Caires, Pfenning and Toninho [3].

### 4.1 DILL

DILL has got its name from the standard Dyadic (or Dual) sequent calculus for Intuitionistic Linear Logic. Dyadic stands for binary relation. Sequent calculus is a way of logical argumentation, where every line within a proof belonging to this calculus is a conditional tautology (or 'sequent'). Intuitionistic Linear Logic emphasizes proofs, where formulas are used as resources to create those proofs. The difference between classical logic and intuitionistic logic is that systems of intuitionistic logic do not include the law of the excluded middle and double negation elimination, which form the basic inference rules in classical logic. All of the previous notions combined results in the basis upon which DILL is defined.

DILL was first introduced by Caires and Pfenning [2] as a type discipline for the $\pi$-calculus. The authors present it as 'the first purely logical account of both the shared and linear features of session types'. Claims are made that this new typing system ensures session fidelity as well as the absence of deadlocks. Session fidelity means that processes follow the communication protocols defined for the channels that are used by those processes. The authors also show that there is a correspondence between so-called cut elimination steps in sequent calculus and simplifications in $\pi$-calculus descriptions.

DILL uses the formulas that intuitionistic linear logic provides and sees them as a form of session types. This way, the connectives that this form of logic uses can be preserved in the syntax of the typing system. The most basic example of a statement in DILL would be $A \multimap B$. This is a linear function that takes an object of type $A$ as input and returns an object of type $B$ as output. In DILL, this is denoted by a session $x$ that first inputs a session channel of type $A$, after which it behaves as $B$. Another basic statement in DILL is $A \otimes B$, which denotes a session $y$ that first outputs a session channel $A$ and then behaves as $B$ again. These two simple statements each consist of exactly two session objects, namely one of type $A$, and one of type $B$. The former is used for the communication, while the latter is used for the continuation of the process. Objects $A$ and $B$ do not interfere with one another, they are only linked in the sense that $B$ happens after $A$ has finished.

Since we want to give a more meaningful example within this typing system, we first present the syntax of this typing system. Suppose $A$ and $B$ are session objects again, then the grammar of the DILL typing system can be written as shown below.

$$A, B \ ::= \ 1 \mid {!A} \mid A \otimes B \mid A \multimap B \mid A \oplus B \mid A \& B$$

The two simple statements that we talked about before, $A \multimap B$ and $A \otimes B$, are each others dual. Where the former receives input on channel $A$, the latter sends output on channel $A$. The grammar also contains 1, which denotes session termination. The type $!A$ specifies a non-session channel that can be used for creating new sessions of type $A$. Then we also have $A \oplus B$ and $A \& B$, which are also dual to one another. The former is used for selecting one of the presented options and the latter is used for offering such options.

To obtain a better understanding of the way processes can be described using this typing discipline, a more concrete example is given. The example contains some person wanting to get a coffee from a coffee machine, and the coffee machine itself. The latter offers two choices, it is possible to buy coffee as well as tea. Thus, both actions require payment, before the drink is dispensed. The coffee and tea machine could be described in DILL as shown below.

$$\text{Dispenser} \ ::= \ (\text{Coin} \multimap (\text{Coffee} \otimes \mathbf{1})) \ \& \ (\text{Coin} \multimap (\text{Tea} \otimes \mathbf{1}))$$

As can be seen, in either choice the input of a coin is needed. After this, based on the choice that the person made, either coffee or tea is dispensed. After dispensing, the session is terminated. The person wanting to purchase a coffee would then be described by the process below. The Dispenser process is assigned to channel $y$ in this description.

$$\text{Person} \ ::= \ y.inl;\ y\langle \text{coin} \rangle . y(\text{coffee}).\mathbf{0}$$

The person selects the left option from the dispenser using the session channel $y$. Then, money is outputted along the channel, after which coffee is inputted along that same channel. After receiving the coffee, the process terminates. This is a very simple example, yet it shows that the DILL discipline can be used to easily specify behavior of such real-world systems.

### 4.2 CLL

For the second system that we analyze, we look at the session type system which is called CLL. CLL corresponds to Classical Linear Logic, which is the basis upon which this system is built. This system makes use of the rules and interpretations of this classical discipline, which leads to the loss of the locality property on shared channels. Using the locality property is the most practical choice for distributed implementations of channels [3]. Locality prevents a process from receiving data on previously received channel names. Non-locality would allow a process to receive data on a shared channel created somewhere else, possibly causing unwanted interference.

To get a more concrete view of this typing system, we present its syntax. Suppose $A$ and $B$ are session objects again, then the grammar of the CLL typing system can be written as shown below.

$$A, B \ ::= \ \perp \mid 1 \mid {!A} \mid {?A} \mid A \otimes B \mid A \parr B \mid A \oplus B \mid A \& B$$

The most important changes in this typing system with respect to DILL are threefold. First, an additional negation operator is added to the syntax ($\perp$). Second, using this negation operator and the fact that the system should have full duality, the 'why-not' exponential connective $?A$ is added. Lastly, the connective of the input session type is changed to $\parr$ instead of $\multimap$. Due to the fact that the syntax is quite similar to that of DILL, it would not be that meaningful to present the example of the coffee machine from before in CLL. The specification would be completely analogous to the one presented in the DILL example.

### 4.3 CP

CP is introduced and presented by Wadler [11]. It uses propositions of classical linear logic to represent session types. CP is inspired from DILL, but unlike DILL, it uses one-sided sequents instead of two-sided ones. This offers a greater simplicity and provides symmetry - with a single construct for output ($\otimes$) and a single construct for input ($\parr$), each dual to the other.

$A, B, C ::=$

| | |
|---|---|
| $X$ | propositional variable |
| $X^{\perp}$ | dual of propositional variable |
| $A \otimes B$ | times - output $A$ then behave as $B$ |
| $A \parr B$ | par - input $A$ then behave as $B$ |
| $A \oplus B$ | plus - select from $A$ or $B$ |
| $A \& B$ | with - offer choice between $A$ or $B$ |
| $!A$ | of course! - server accept |
| $?A$ | why not? - client request |
| $\exists X.B$ | existential - send a type |
| $\forall X.B$ | universal - receive a type |
| $1$ | unit for $\otimes$ |
| $\perp$ | unit for $\parr$ |
| $0$ | unit for $\oplus$ |
| $\top$ | unit for $\&$ |

Fig. 2. Grammar for propositions in CP that are interpreted as session types (reproduced from Wadler [11])

The grammar for the propositions that are interpreted as session types in CP is presented in Figure 2. For every propositional variable $X$, its dual is constructed as $X^{\perp}$. The aspect of duality plays a crucial role. It ensures that requests for input match the offers for output, requests for selection match the available offers for choice, and sending of types matches the receiving of types. In other words, it assures the compatibility between communicating processes. Propositions are composed from multiplicatives ($\otimes, \parr$), additives ($\oplus, \&$), exponentials ($!, ?$), second-order quantifiers ($\exists, \forall$), and units ($1, \perp, 0, \top$).

### 4.4 GV

The work on GV is extended by Gay and Vasconcelos [7], where they provide the formalization of a system in which the send and receive operations are buffered, instead of relying on synchronization. Wadler [11] shows that GV and CP are semantically equivalent by presenting a translation between the two session type frameworks.

To give a sense of this session framework, we explore its syntax by first taking a look at the grammar, presented in Figure 3.

$$
\begin{aligned}
T &::= S \mid D \mid T \otimes T \mid T \to T \mid \langle S\rangle^{r} \mid \langle S\rangle^{a} \mid \langle S, S'\rangle^{a} \\
S &::= end \mid ?T.S \mid !T.S \mid \&\langle l_i : S_i\rangle_{i\in I} \mid \oplus\langle l_i : S_i\rangle_{i\in I} \mid X \mid \mu X.S \\
D &::= Unit \\
B &::= T \mid l
\end{aligned}
$$

Fig. 3. Syntax of general types and session types in GV (reproduced from Gay and Vasconcelos [7])

Session types operate on channels. The grammar in Figure 3 shows that there are multiple types of channels a session can have. First, *end* represents a channel type that does not allow for further communication to occur. The duals $!T.S$ and $?T.S$ determine types of channels from which a message of type $T$ can be sent an received respectively. After the completion of this action, the channel becomes of the type specified by $S$. Additionally, $\&\langle l_i : S_i\rangle_{i\in I}$ and its dual $\oplus\langle l_i : S_i\rangle_{i\in I}$ allow for offering a choice of selection between the distinct labels $l_i$ and making a selection respectively, by sending one of the labels $l_i$. Again, these are duals and the subsequent behavior is described by the appended session type $S$. Last but not least, recursive session types are allowed by the construction $\mu X.S$ as long as the recursive element $X$ is contractive.

We give a simple example to show how GV can be used to model real scenarios, inspired by Gay and Vasconcelos [7]:

$$Restaurant = \&\langle add : ?Dish.Restaurant, buy : ?Card.?Address.\mathbf{end}\rangle$$

The operator $\&$ indicates that the restaurant offers two choices: adding dishes to our order and buying the order. After add, the restaurant receives data of type Dish and then returns to its original state. The buy action ends with the restaurant receiving the payment and delivery details of the purchaser. On **end** the interaction is stopped. On the other side, the shopper also behaves according to a protocol, which is dual to the one presented above:

$$Shopper = \oplus\langle add : !Dish.Shopper, buy : !Card.!Address.\mathbf{end}\rangle$$

As we can see, the applicability of session types is extremely versatile. It includes not only network protocols, but also business processes and can even be attributed to operating system services [7].

## 5 Extensions

The concept of session types is being actively developed. Different applications require different functionalities, therefore, different extensions to the original formulation of session types are being researched. In this section, we look at two different approaches of extending the binary session types: the first is focused on a solution in which specifying properties of the data that is being exchanged is added, while the second is a way to allow for more than two peers two establish and participate in a communication session.

### 5.1 Dependent session types

As previously mentioned, there are different kinds of session types. Dependent session types are one of those. These session types allow for the description of exchanged data instead of only the input and output behaviour of processes. This can be exploited as to express proof-carrying communication [8]. Dependent session type systems are layered ones, with concurrent communication at the outer level and a dependently typed functional language at the inner level.

Proof-carrying code can be modelled by these dependent session types, because dependent type theories correlate proofs with programs. The application of proof-carrying code is done by using digitally signed certificates that can be used as proof objects, such that the complete proof does not have to be sent along with the program in question. This way, the complete proofs do not have the be checked upon receiving them, which can be considerably more work than simply verifying the corresponding certificate. These certificates are generated by a third party that creates the proof for the program, and when it satisfies the needed properties, the certificate is coupled to that program. Determining which code is accompanied by the corresponding proof, which is accompanied by a certificate, and which is trusted without a doubt, is still to be determined [8].

In order to make it more clear what such a dependent session type system looks like, an example is given of a session that accepts code for a function on natural numbers. It also gives a proof that this function is strictly increasing:

$$\forall f : nat \to nat. \forall p : (\Pi x : nat. f(x) > x).\mathbf{1}$$

A proof object is sent along with the function $f$. This proof shows that the function is strictly increasing. Checking complex proofs may be delegated to trusted parties, which sign a certificate when they checked the proof successfully. By introducing such a certificate, the proof itself does not need to be sent along anymore. The specification in this dependent session type system may then look like:

$$\forall f : nat \to nat. \forall p : \diamond_{cert}\big[\Pi x : nat. f(x) > x\big].\mathbf{1}$$

Receivers of this verified function only get the trusted, signed certificate and not the complete proof. This saves them the trouble of needing to check the proofs. In the context of software and computation, this lack of need for checking proofs is valued from the perspective of computational efficiency. This is because it allows for performing less computations while still achieving similar levels of security and assurances of the correctness of the communication and data of the function or program we intend to run.

### 5.2 Multiparty session types

Until now we have looked at the simplest form of session types - binary session types - which are used to establish communications between exactly two peers. For real world interactions however, these are not always sufficient. A conversation between processes can involve, in general, any number of processes. To be able to express this type of communication, multiparty session types have been introduced.

The three components that are necessary for defining multiparty session type systems as presented in the paper of Coppo *et al.* [4] are:

1. There exists a *global type* at the highest level of abstraction that describes interactions between peers. It resembles a protocol, but one which is neutral with respect to the interacting parties, while still listing the types of messages that are allowed to be exchanged.
2. The behaviour of the peers - the lowest level of abstraction - is presented through process specifications (expressed in $\pi$-calculus or similar).
3. The two levels are connected with *local types* which represent interpretations of the global type from the perspective of each peer. These types are obtained upon a *projection operation* applied to the global type.

The above listed properties are thus translated into the grammar for the multiparty session types, presented in Figure 4. From this we notice that one of the main differences between multiparty session type systems and binary session type systems is the required inclusion of the reference to the participant at communication.

$$
\begin{array}{lll}
 & S ::= bool \mid ... \mid G & \textit{sorts} \\
 & U ::= S \mid T & \textit{exchange types} \\
\textit{Global types} & & \\
 & G ::= p \to q : \langle S \rangle.G & \textit{value exchange} \\
 & \quad \mid p \to q : \langle T \rangle.G & \textit{channel exchange} \\
 & \quad \mid p \to q : \{l_i : G_i\}_{i \in I} & \textit{branching} \\
 & \quad \mid \mu t.G \mid t \mid end & \textit{recursion and end} \\
\textit{Session types} & & \\
 & T ::= !\langle p, S \rangle.T & \textit{send value} \\
 & \quad \mid !\langle p, T \rangle.T & \textit{send channel} \\
 & \quad \mid ?(p, U).T & \textit{receive} \\
 & \quad \mid \oplus\langle p, \{l_i : T_i\}_{i \in I} \rangle & \textit{selection} \\
 & \quad \mid \&(p, \{l_i : T_i\}_{i \in I}) & \textit{branching} \\
 & \quad \mid \mu t.T \mid t \mid end & \textit{recursion and end}
\end{array}
$$

Fig. 4. Syntax of multiparty session types (reproduced from [4])

## 6 Discussion

We start our discussion by first defining the expressiveness characteristics that we have identified while reviewing the four session type systems and the two extensions. We then continue on to relating them with the corresponding session type frameworks.

### 6.1 Forward Definitions

A race condition occurs whenever two or more processes, that can access shared data, try to change it at the same time. In other words, the output of the system is dependent on the sequence or timing of the involved events. The implications from such traits can be extremely damaging since it leads to unpredictable and possibly wrong behaviour of the program.

A deadlock represents a state in which each member of a group is waiting indefinitely for another member to take action. Therefore, mutual unavailability of resources causes processes to block and await the release of those resources, hence the program effectively stops. The possibility of such circumstances within a system makes it again - unpredictable and unreliable and it is no longer possible to make any claims regarding the global progress of a program.

Session fidelity describes how closely the processes follow the communication protocols for the session channels. If processes do not satisfy this property, unexpected and thus unwanted behaviour can be the result of this. Data that is meant to be received by one session could be received by another session instead, leaving the former in an awaiting state and providing the latter with data that is not relevant for that session. This could lead to errors in execution of such processes.

Polymorphism allows for describing interfaces by abstracting on the used types. It improves the expressiveness of a language since it grants the ability to define the behavior of processes without regards to the types. This implies that session types acting as protocols could be interchanged, thus aiding the versatility of the program.

Subtyping introduces the ability to extend functionality while maintaining compatibility with the old infrastructure. This is a concept often encountered in the software realm since it gives the ability to create hierarchies of types. This in turn provides the benefit of reusability. In the context of protocols, subtyping gives the advantage of versatility - generic protocols could be easily extended to specific needs.

Recursive types are types that can contain values of the same type. In the context of session types this translates to recursive protocols. These are useful to define constructs where different branches can follow different protocols.

### 6.2 Analysis

In the DILL typing system, a session either terminates following **1** or it becomes a spawn point for new sessions following the $!A$ operator. This is a consequence of the interpretation of the logic the system is based upon. Not all session types have this interpretation. Parallel composition however is included in most session type systems. Together with certain rules of DILL, it allows for ensuring global progress. This is not the case with traditional session type systems, where multiple open sessions can be present [2].

DILL also contains the subject reduction property, which leads to session fidelity. Subject reduction means that evaluating expressions does not cause their type to change. Even when there are multiple communicating sessions at play, it ensures global progress. These two properties lead to the type soundness of the typing system, in this case DILL. All of this means that communication protocols are followed very closely and there is an absence of deadlock in the specifications in this typing system.

The CLL typing system is quite similar to DILL in certain aspects. The syntax of the two is analogous, resulting in the fact that the only difference between a simple specification in DILL and the same specification in CLL, is the replacement of the connective for input. Of course, when dealing with more complicated specifications, clear distinctions between the two become visible.

Previously proposed session types do not all enforce the locality property of shared channels, so CLL is not unique in this aspect. Non-locality is not expressible in DILL, and thus moving from a classical session type system to an intuitionistic one enforces this locality property.

In CP, two concurrently executing processes $P$ and $Q$ are disjoint. There is no entangling between the channels on which the two communicate. This guarantees CP to be free from races and deadlock. Additionally, this is also supported by the property that any process can reduce until it needs to perform an external communication - operation identified as top-level cut elimination.

For any propositional variable in CP, its dual can be constructed (see Figure 2): input is dual to output, selection is dual to choice. This feature determines the ability to create compatible protocols, in which the communicating processes are bound to be able to exchange the necessary information at any point during their interaction.

The correspondence for the quantifiers $\exists$ and $\forall$ with polymorphic $\lambda$-calculus on type application and type abstraction, or to sending and receiving types in $\pi$-calculus, deems CP able to support impredicative polymorphism. A polymorphic type may thus be instantiated by a polymorphic type. Specific care needs to be taken upon formulation to ensure that termination remains valid.

The session type of a server that repeatedly accepts an $A$ is described with the operator $!A$ - 'of course!'. The session type of a collection of clients where each may request an $A$ is described with $?A$ - 'why not?'. A server has to be impartial and provide the same service to any incoming client. On the other hand, clients could pass different requests to one server. From this, the above description of servers defines uniform behaviour while the description for clients accumulates diverse behaviours. Resulting from this is the fact that CP gives the ability to add replicated servers with multiple clients.

In the original definition of GV, Gay and Vasconcelos [7] describe a type system that does not guarantee deadlock-freedom as it is based on asynchronous buffered communication. It allows for two threads to be blocked waiting for input where the matching outputs are behind the inputs in the same two threads. They continue on to claiming that even though restrictions to the language could be applied to achieve the above mentioned features, interleaving of sessions is important for expressiveness. The modified GV, used in the translation from Wadler [11] becomes free of race conditions and deadlocks. This is however achieved by restricting the semantics to a subset of the original definition.

Even without the guarantee of strong progress, GV is still bundled with support for recursive functions, recursive session types, subtyping, and offers a reliable foundation for further developments such as polymorphism and object-orientation. It is however worth to note that well-typed programs terminate unless they explicitly resort to non-logical features, such as general recursion. Thus, adding recursive terms or recursive session types may remove the property that all programs terminate.

By making use of dependent session types, Pfenning, Caires and Toninho [8] present a way of providing proofs along with the data that is being exchanged. They continue on to say that these proofs could even be omitted, when a trusted third party verifies the data and its proof, then signs a certificate claiming this, and sending it along with the data instead of the complete proof. This idea provides a way of making the verification of exchanged data (for example code) far more efficient in terms of computation needed, especially when dealing with complex data that would take quite long to verify successfully. Proof-carrying code ensures progress and session fidelity, while making use of polymorphism as well.

The efforts from Caires and Perez [1] introduce a multiparty session type system that is capable of parametric polymorphism, or in the context of their work - behavioural genericity. Additionally, basing their system on binary session types and by acquiring the information about the sequence of actions of processes using a process extracted from a global type, allows them to build a system that is also deadlock free.

## 7 Conclusion and Future Work

We have analyzed several session type systems in this paper, extracting properties that are characteristic for these typing systems. Before these systems are presented, we introduce the concept of session types and the basis upon which these are built. A selection of four different session typing systems was made to conduct this research, of which all of them are proposed by other authors within this field of research. Additionally, several small examples of specification within the typing systems are given and explained. Two extensions of typing systems are presented as well. The different typing systems and some extensions of them are discussed further by arguing about their main characteristics.

Most of the papers taken as reference material for this analysis of session types, session type systems, and extensions of them are quite recent, showing that the concept of session type systems is one that was born not so long ago. This is not very surprising, since the needs and uses for them are also things that only became apparent over the last decade. This is of course due to the fact that distributed concurrent systems become more widely used every year, which means that they become almost essential for current day society. This leads to the need for correct communication with respect to these systems, since users expect them to run without faults.

Future work may include making improvements of the session type systems that are presented here, based on the shortcomings that they may have in the current situation. For example, DILL does not yet include the notion of recursive types, while some typing systems presented by other authors within the field do. Adding this to DILL may not be that difficult on itself, yet by doing this, the connections with the logic that the system was based upon disappear. By studying inductive session types, and using them in order to add the notion of recursive types to the typing system, these connections with the logic may be preserved.

Another possibility of future work may be to research extensions of the CP typing system, which would allow for this system to become more powerful by using more aspects of the $\pi$-calculus. This would make the typing system more widely applicable, increasing its uses and therefore the interest in the system.

## Acknowledgements

We wish to thank Jorge A. Pérez for acting as expert reviewer on this paper and providing valuable feedback, suggestions and corrections. We also wish to thank our colleagues, Enrico Werkema and Marios Lykiardopoulos, for their input and observations.

## References

[1] L. Caires and J. A. Pérez. Multiparty session types within a canonical binary theory, and beyond. In *International Conference on Formal Techniques for Distributed Objects, Components, and Systems*, pages 74–95. Springer, 2016.

[2] L. Caires and F. Pfenning. Session types as intuitionistic linear propositions. In *International Conference on Concurrency Theory*, pages 222–236. Springer, 2010.

[3] L. Caires, F. Pfenning, and B. Toninho. Linear logic propositions as session types. *Mathematical Structures in Computer Science*, 26(3):367–423, 2016.

[4] M. Coppo, M. Dezani-Ciancaglini, L. Padovani, and N. Yoshida. *A Gentle Introduction to Multiparty Asynchronous Session Types*, volume 9104 of *LNCS*, pages 146–178. Springer, 2015.

[5] O. Dardha, E. Giachino, and D. Sangiorgi. Session types revisited. *Information and Computation*, 256:253 – 286, 2017.

[6] S. J. Gay. Subtyping supports safe session substitution. In *A List of Successes That Can Change the World*, pages 95–108. Springer, 2016.

[7] S. J. Gay and V. T. Vasconcelos. Linear type theory for asynchronous session types. *Journal of Functional Programming*, 20(1):19–50, 2010.

[8] F. Pfenning, L. Caires, and B. Toninho. Proof-carrying code in a session-typed process calculus. In *International Conference on Certified Programs and Proofs*, pages 21–36. Springer, 2011.

[9] D. Sangiorgi and D. Walker. On barbed equivalences in $\pi$-calculus. In *International Conference on Concurrency Theory*, pages 292–304. Springer, 2001.

[10] V. T. Vasconcelos. Fundamentals of session types. *Inf. Comput.*, 217:52–70, Aug. 2012.

[11] P. Wadler. Propositions as sessions. *Journal of Functional Programming*, 24(2-3):384–418, 2014.

[12] P. Wadler. Propositions as types. *Communications of the ACM*, 58(12):75–84, 2015.

# Reversible Operating Systems
## An overview of the state of the art and its future challenges

Thijs van der Knaap, Luc van den Brand

**Abstract**—The viability of a reversible operating system in the context of current developments in reversible computing is determined. Conventional computing is inherently irreversible: once a value has been written, the information required to revert the operation is lost. Reversible computing only allows for instructions that have a reversing counterpart, thus allowing a machine to return to any stable state by executing its instructions in a reverse order. With this notion in mind, a reversible operating system would be a system that provides a platform-independent framework for developing reversible applications.

By examining the hardware, scheduling system, services and general environment of a conventional micro-kernel OS architecture, the risks and benefits of implementing reversible solutions to each component will be discussed.

We conclude with a summary describing the open problems that need to be solved for each component of the operating system.

**Index Terms**—Parallel architectures, operating system, reversibility, programming.

## 1 Introduction

Reversible computing allows for unwinding operations: executing the negation of each operation such that a machine is capable of reverting to any previous state if so desired. Reversible computing is useful in a variety of cases, among which is the increased energy efficiency of hardware implementations [11] and the recoverability of distributed systems [5]. Although both topics are well discussed, less work has been done on implementing solutions that abstract the fundamental layers into a general purpose framework for multi-process reversible applications. Thus, reflecting on the current developments in the different branches of reversible computing, we will determine the viability of a reversible Operating System (OS).

To give an initial view on the current state and solutions in the science of reversible computing, some related work on Causal Consistent Reversibility and Reversible CCS is discussed in Section 2. In Section 3 the general risks and benefits in developing a reversible OS are discussed. These risks and benefits are composed of the core components from which such a system would need to be constructed, these are discussed in Section 4. Further inspection of these components is continued in Subsection 4.1 through 4.5. In Section 5, we will conclude with a short discussion of the viability of a reversible OS and make a proposal for further research.

## 2 Related work

Currently, most research performed in the field of reversibility is aimed at finding formal solutions to acquire correct reversibility. Two very important concepts in this field are Causal Consistent Reversibility and Reversible CCS. The paragraphs below provide a brief overview of both concepts.

**Causal consistent reversibility** is defined as: "*any action can be undone, provided that all its consequences (if any) are undone beforehand.*" [12]. This definition seems to be very straightforward, but in reality it is quite hard to keep track of all consequences of a function. In the paper cited above, the authors state two properties that are vital to ensure proper reversibility, namely the **Loop Lemma** and **Causal consistency**.

The Loop Lemma simply states that every transition should also have a reverse transition. The builder of a reversible system should be warned though, without proper control a program can can get stuck in a live-lock: performing and undoing the same operation continuously.

Causal consistency is a property that if acquired ensures that a reverse will undo all behaviour. As the name suggests it is a relaxation on normal consistency, since we only care about the results of the process. Two processes are causal consistent if and only if it holds that any transition starting in the same state always terminates in the same final state. This makes the two processes indistinguishable for an external observer, allowing the scope of our interest, reversibility, to not care about which of the two transitions was used. It is important to ensure that transitions are causal consistent to reduce the amount of bookkeeping that has to be performed for reversible systems and to prove their correctness.

**Reversible CCS** (RCCS) [7] gives a formal language to define systems that are able to backtrack. The basis of RCCS can be found in Milner's Calculus of Communicating Systems (CCS), which is a process calculus that models the communication in concurrent systems [13]. RCCS aims at finding a balance between allowing for maximal concurrency while still retaining correct reversibility. If you simply let a concurrent program run and only reverse a single thread, the resulting reverse will probably not be correct. If only one thread at a time is allowed to run, it must be reversible, however then one simply obtains a synchronous program. RCCS tries to find a balance between the two previously explained cases by allowing concurrent programs to run, whilst acquiring enough information to allow for reversibility. The key feature of RCCS is that it does not only store the memory stack of every running thread, but that it also keeps track of the communication between these threads. This knowledge of communication allows RCCS to reverse a thread correctly, since this reversal is able to propagate the reverse to other threads that were synchronized or forked.

## 3 Risks and Benefits

A reversible OS can be seen as a machine with the usual *"undo button"* integrated into its core fundamentals. Such a feature can be quite useful to escape out of failed states, as unexpected termination of applications often lead to loss of information or require time-consuming recovery procedures. An OS that can easily revert to a stable state can therefore be beneficial to both the average consumer and the system engineer requiring automated recovery in a fault-tolerant system [15].

As is common when engineering computer systems, implementing a solution on a **hardware** level can often lead to the highest increase in performance. Take for example the increase in performance gained when 3D computer graphics transitioned from software rendering to hardware rendering, or the more recent performance increase gained by running hardware-level Artificial Intelligence [10]. Reversible computing components can additionally be more energy efficient than their irreversible counterparts [11].

Implementing applications in a comparatively low level of abstrac-

- *Thijs van der Knaap is with the University of Groningen, E-mail: t.v.d.knaap@student.rug.nl.*
- *Luc van den Brand is with the University of Groningen, E-mail: l.a.h.van.den.brand@student.rug.nl.*

tion can however be more error-prone than implementing the same solution on a higher level [3]. Developing a software layer that neatly abstracts the low-level hardware components can therefore increase the ease at which future reversible solutions are developed. If one were to design a reversible OS from scratch, it might thus be beneficial to build it on top of an already reversible architecture. An alternative to implementing the OS on reversible hardware is the use of **virtualization**, which allows for implementing the system on current hardware but will inevitably require an overhead to simulate reversible instructions.

A core feature of modern OS's is the support for multiple applications to be executed concurrently. Even a single application is often allowed to spawn threads that represent concurrent branches of the same process. Reversing parallel systems can be much more challenging than reversing their sequential counterparts, as their arbitrary **scheduling** can cause ambiguity for which operations should be reverted in which order. Causal Consistent Reversibility (Explained in Section 2) provides some helpful guidelines in this matter, dictating that an operation can only be reverted if every operation that depends on its results is reverted first.

A core component of OS's is their collection of integrated **services** that provide simplified abstractions to complex procedures commonly required by applications, such as reading and writing to files, reading device input and producing device output. Reversing some of these services can however be non-trivial, requiring interpretations on irreversible operations.

Software compatible with the set of machine instructions provided by the reversible OS would still need to be developed. In other words, **compilers** would need to be developed for reversible programming languages. These languages, in contrast to the more abstract languages such as RCCS, are meant to allow for the implementation of reversible software solutions. A collection of these languages is available, ranging from functional to imperative paradigms.

## 4 Components of a reversible OS

Generally speaking, OS's are a layer of indirection between the physical computing infrastructure and the more abstract commands of one or more applications. OS's allow applications to run on more than one machine as long as more than one machine supports the same OS, effectively decoupling application software from hardware. In addition, an OS usually provides a set of complex instructions that are too difficult to implement on a hardware level but are often required by non-trivial applications [17].

An OS can itself be split into multiple components, as can be seen in Figure 1. The *kernel* is the only component that is allowed to run in *kernel mode*. When an application is in kernel mode, it can communicate with the hardware layer directly. All other applications run in *user mode*, which requires all instructions to be interpreted and managed by the kernel before execution. The distinction of kernel mode and user mode allows the OS to restrict control over potentially dangerous instructions, increasing security and reducing fatal failures.

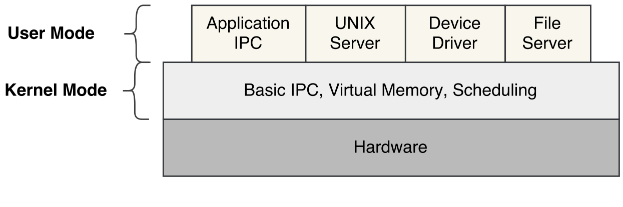


Figure 1. The general architecture of a Micro-Kernel OS.

The figure above describes what is also known a *micro-kernel architecture* [17]. This refers to the design philosophy of having only the critical components of an OS run in kernel mode, while all other components are implemented as applications that can be relaunched in case of failure. In our case, this architecture also allows for *reverting* the separate components of the OS individually, increasing the fault-tolerance of the OS and reducing the complexity of enforcing Causal Consistent behavior.

The included benefit of this architecture is that it best serves as a guiding map for the core challenges faced when implementing a reversible OS:

**Hardware Architecture** - The physical components on which the OS will function need to be reversible to enable operations to be reversed. An alternative is a strong layer of virtualization allowing irreversible hardware to simulate the properties of reversibility.

**Scheduling** - Even on single-core processors, applications can be made to execute in seeming concurrency due to the sharing of execution time. This concurrency would need to be Causal Consistent in order to support proper deterministic reversibility.

**Services and Drivers** - Implementing reversibility for device drivers is a non-trivial problem, as actions such as producing sound are often impossible to revert.

**Compilers** - In order for an OS to be of any use, it must support the development of applications that can execute in its environment. This requires the compilation of reversible languages to the OS's supported machine instructions, and the potential definition of libraries that further increase the ease of development.

### 4.1 Hardware Architecture

A proposal for one of the first reversible computing components comes from Carlin James Vieri, who defined an architecture for a fully reversible microprocessor called Pendulum [19]. The Pendulum Assembly Language (PAL) defined for this microprocessor provides instructions that allow for reverse computation. More recent developments feature architectures such as Bob and its instruction set BobISA [18], which claims to be both "practical and feasible" to implement. Currently, the primary drive behind implementing these systems is the low amount of energy required to execute instructions on a system created from reversible components [8, 11]. This would be especially beneficial for battery powered systems like laptops and phones.

Although these architectures provide the capability to execute its atomic operations in reverse, they inherently only support *sequential* reversibility. Any application developed in languages such as PAL or BobISA would still need to be designed with reversibility in mind, requiring extra care to support notions such as Causal Consistent reversibility when handling multiple concurrent processes (see Section 4.3).

At the moment of writing, none of these theoretical architectures have been implemented. Calls for doing so have been made [9], but until they are answered a reversible OS would need to run on conventional irreversible computing hardware.

### 4.2 Virtualization

Using virtualization to acquire a reversible OS brings forth similar levels of reversibility as implementing it on actual reversible hardware. There is however one significant disadvantage: where the hardware simply has operations for forward and backward computations, the virtual machine has to simulate this behaviour. The virtual layer will be visible to the rest of the OS as fully reversible hardware. All the actions that the virtual machine offers therefore need to have an undo operation. These undo operations are implemented in the virtual machine using normal non-reversible hardware.

Because all the system calls pass through an indirection layer, where bookkeeping has to be performed to ensure reversibility, the system will suffer in performance. The decrease in performance for a non-reversible virtual machine can be significantly reduced [14, 1], however it can not be eliminated. Add to this the inevitable bookkeeping, and a real problem presents itself.

The significant advantage that is gained from using virtualization, in contrast to reversible hardware, is that the virtual layer will be able to run on current non-reversible hardware. This can reduce the adoption time of reversible systems dramatically.

Multiple efforts have already been made in the field of sequential reversible virtualization [16, 4]. They mostly are supporting one heavily restricted programming language and do not support concurrency. It is nice to see that attempts are made in this field, but currently it is far from mature. It is essential for a viable virtual machine to eliminate the specific programming language dependency and to incorporate concurrent systems.

The reduced power consumption that is achieved by using reversible hardware is not present in its virtualized form. The reverse process is simulated using normal operations, requiring more energy when compared to traditional irreversible solutions in order to process instructions through a virtual machine.

### 4.3 Scheduling

In an environment in which multiple processes can share the same execution environment, we must take care to ensure Causal Consistent behavior. The system must therefore keep track of which instructions were caused by which processes, such that it can properly revert the consequences of an operation before the operation itself is reverted.

To better understand the true nature of this problem, imagine two scheduled processes running on a single-core processor (see Figure 2). Process 1 tries to continuously write the letter ‘A’ to a file, whilst Process 2 reads from this very same file and writes a letter ’B’ if the last character is an ’A’. Over time, Process 1 and 2 alternate in using the processor, performing their tasks when allotted time to do so. It is clear to see that the behavior of Process 2 depends on the behavior of Process 1, together generating a set of strings that can strictly be defined using the regular expression $(A(B|\emptyset))^{\star}$.

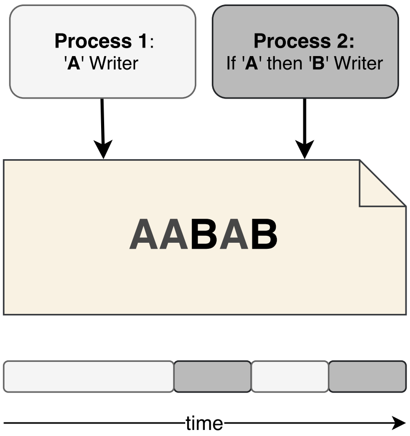


Figure 2. An example case of single-core process scheduling.

Now, imagine a naive approach of implementing reversibility which assumes that each process executes in isolation. In the example given above, reversing Process 1 a single step back requires it to undo the the write of the letter ‘A’, resulting in the string ‘AABB’. This state is incorrect, as it does not fit in our previously stated set of strings that our derived regular expression defines. Instead of reversing the machine into a previous state, we have transitioned the machine into a state that is entirely undefined within the context of the system as a whole.

Luckily, if we strictly enforce inter-process communication through system calls, we are able to record inter-process synchronization and thus should have enough information to reverse multi-process interactions in a manner that is well defined. This is another benefit of the micro-kernel architecture, as it inherently disallows an application to perform actions on a hardware level, and thus requires system calls for inter-process communication [17].

### 4.4 Reversible Services

Looking at the micro-kernel architecture shown in Figure 1 we can see that an OS also contains elements like the file server and device drivers. These programs are responsible for communicating to the world "outside" of the CPU. It is possible to make a totally reversible OS without having reversible services. As an example, a created file can be reversed by deleting the named file. But this brings forth a major disadvantage: the builder of the central system has to think about the reversibility of every type of driver. It would be very beneficial if driver builders would take the responsibility to add reverse calls to their API.

There are also drivers that are not truly reversible, for example reversing a printer-driver does not make a printed page go blank nor makes it magically fly back into the paper tray. Still, reversibility can be applied in this case to ensure that the printer-driver at least returns to the state before it printed the page. This form of state-reversibility is still only applicable for a subset of all device drivers. For instance, reversing a graphics driver by recording its internal state would result in an unpractical amount of memory overhead.

Examples nor research concerning the field of reversible services are few at best, but many formal solutions found in other areas can be applied here. As stated above it is not essential for proper reversibility of the CPU to have reversible services, but it will allow the system to be a lot more efficient and manageable. Therefore it is essential for promoting the proper adoption of a reversible OS.

### 4.5 Compilers

For an OS to support applications written in more complex languages than assembly, we would inherently need a compiler that can convert one reversible language to another. Implementations exist for the high-level reversible language Janus, converting it into the reversible assembly language PISA [2]. Compiling Janus to PISA creates no more than a constant overhead, similar to the classical compilation of irreversible languages.

Other less obvious efforts are made in constructing reversible compilers at the domain of debuggers. For debugging it is very useful to be able to perform a step back in execution. Some debuggers have implemented this reversibility as a simulation, where every step back actually runs the whole program again except for the last step. Luckily, there are also debuggers that provide proper reversibility, for example the very well known GNU Debugger (GDB). This shows that acquiring reversibility for an originally non reversible language is possible, although it will significantly decrease the performance of the compiled programs [6].

Previously mentioned compilers all assume that a to-be-compiled program executes sequentially. For small projects this might not be a problem, but to acquire full power reversible concurrency must be acquired. Section 4.3 explains the problem and gives an insight into the current state of the art.

To promote the adoption of reversible programming, libraries should be created for popular programming languages like Python and Java. This should reduce the adoption time. The final goal is to construct a new programming language like Janus that supports concurrency and reversibility. The advantage of having a dedicated programming language is that it can enforce constructs that are beneficial for reversibility, for example allowing only functions without side effects.

## 5 Conclusion

Theoretical solutions exist that allow for reversible computing on a hardware level. When these are implemented in practice, a reversible OS should prove useful in providing a framework for implementing a variety of reversible applications in a higher level of abstraction than the hardware-dependent machine instructions.

However, for a reversible OS to be viable with existing hardware, a virtualization level is required that simulates a reversible architecture using irreversible instructions. In addition to the significant increase in overhead, current solutions do not support concurrent processing and are limited to research on a single instruction set.

Single-core virtualization should still allow for time-scheduling multiple processes, and thus requires an implementation of scheduling that enforces causal consistency.

The services provided by common OS's are not always easy to reverse, as they might act on processes outside of the controlled environment of the processor. Their inclusion in the OS is not strictly required, and omitting them can ensure that the entire OS is both deterministic and reversible. Still, including them increases the ease of development for applications that run on the system, therefore increasing its potential adoption.

Finally, work must be done to develop compilers and libraries that support the machine instructions provided by the OS in order to support the development of high-level reversible applications.

To conclude on our analysis as discussed above, we do not think it is currently viable, or even possible to make a reversible OS. However, as can be seen in the Risk and Benefits (Section 3), in specific use-cases a reversible OS can lead to many benefits. Therefore we would like to encourage further research on reversible micro-kernel architectures or its components.

If such a system would be developed, we would like to propose a light-hearted tribute to Unix (the grandfather of micro-kernel OS's) through the act of naming this reversible OS *Xinu*.

## Acknowledgements

The authors wish to thank Jorge A. Perez for his support and guidance in everything related to reversible computing as his very constructive review of our initial draft. Both Alexander Lukjanenkovs as Patrick Vogel deserves recognition as well, as their initial feedback helped us spot many typos and irregularities.

## References

[1] I. Ahmad, J. M. Anderson, A. M. Holler, R. Kambo, and V. Makhija. An analysis of disk performance in vmware esx server virtual machines. In *Workload Characterization, 2003. WWC-6. 2003 IEEE International Workshop on*, pages 65–76. IEEE, 2003.

[2] H. B. Axelsen. Clean translation of an imperative reversible programming language. In *International Conference on Compiler Construction*, pages 144–163. Springer, 2011.

[3] P. Bhattacharya and I. Neamtiu. Assessing programming language impact on development and maintenance: A study on c and c++. In *Proceedings of the 33rd International Conference on Software Engineering*, pages 171–180. ACM, 2011.

[4] P. G. Bishop. Using reversible computing to achieve fail-safety. In *Software Reliability Engineering, 1997. Proceedings., The Eighth International Symposium on*, pages 182–191. IEEE, 1997.

[5] G. Brown and A. Sabry. Reversible communicating processes. *arXiv preprint arXiv:1602.03594*, 2016.

[6] S.-K. Chen, W. K. Fuchs, and J.-Y. Chung. Reversible debugging using program instrumentation. *IEEE Transactions on Software Engineering*, 27(8):715–727, Aug 2001.

[7] V. Danos and J. Krivine. Reversible communicating systems. In *International Conference on Concurrency Theory*, pages 292–307. Springer, 2004.

[8] A. De Vos. Reversible computer hardware. *electronic notes in theoretical computer science*, 253(6):17–22, 2010.

[9] M. P. Frank. The future of computing depends on making it reversible. spectrum.ieee.org/computing/hardware/the-future-of-computing-depends-on-making-it-reversible, 2017.

[10] N. P. Jouppi, C. Young, N. Patil, D. Patterson, G. Agrawal, R. Bajwa, S. Bates, S. Bhatia, N. Boden, A. Borchers, et al. In-datacenter performance analysis of a tensor processing unit. In *Proceedings of the 44th Annual International Symposium on Computer Architecture*, pages 1–12. ACM, 2017.

[11] R. Landauer. Irreversibility and heat generation in the computing process. *IBM journal of research and development*, 5(3):183–191, 1961.

[12] I. Lanese, C. A. Mezzina, F. Tiezzi, et al. Causal-consistent reversibility. *Bulletin of EATCS*, 3(114), 2014.

[13] K. G. L. Luca Aceto, Anna Ingólfsdóttir and A. Jiri Srba. *Reactive Systems: Modelling, Specification and Verification*. Cambridge University Press, 2007.

[14] A. Menon, J. R. Santos, Y. Turner, G. J. Janakiraman, and W. Zwaenepoel. Diagnosing performance overheads in the xen virtual machine environment. In *Proceedings of the 1st ACM/USENIX international conference on Virtual execution environments*, pages 13–23. ACM, 2005.

[15] D. Patterson, A. Brown, P. Broadwell, G. Candea, M. Chen, J. Cutler, P. Enriquez, A. Fox, E. Kiciman, M. Merzbacher, et al. Recovery-oriented computing (roc): Motivation, definition, techniques, and case studies. Technical report, Technical Report UCB//CSD-02-1175, UC Berkeley Computer Science, 2002.

[16] B. Stoddart, R. Lynas, and F. Zeyda. A virtual machine for supporting reversible probabilistic guarded command languages. *Electronic Notes in Theoretical Computer Science*, 253(6):33 – 56, 2010. Proceedings of the Workshop on Reversible Computation (RC 2009).

[17] A. S. Tanenbaum. *Modern operating system*. Pearson Education, Inc, 2009.

[18] M. K. Thomsen, H. B. Axelsen, and R. Glück. A reversible processor architecture and its reversible logic design. In *International Workshop on Reversible Computation*, pages 30–42. Springer, 2011.

[19] C. J. Vieri. *Reversible computer engineering and architecture*. PhD thesis, Massachusetts Institute of Technology, 1999.

# Formal Verification of Sorting Algorithms

Dimitris Laskaratos and Bogdan Petre

**Abstract**— A number of studies have been conducted with the purpose of using formal methods to prove the correctness of well known and widely used sorting algorithms. We analyze some of these studies, which discuss the formal verification of Counting Sort, Radix Sort, TimSort and Dual Pivot Quicksort. We provide a description of KeY, the tool used in these studies to formally verify the algorithms. We also present the most important aspects of each of the 4 algorithms and highlight the steps taken in their analysis with KeY.
The results for two of the studies show formal proofs of correctness of the Java implementations of Counting Sort, Radix Sort and Dual Pivot Quicksort. A third study documents a bug in the OpenJDK implementation of TimSort that resulted in an error being thrown for certain input arrays. It also provides a fix to the bug and formally proves that the updated implementation is correct.

**Index Terms**—KeY, formal methods, Counting Sort, Radix Sort, TimSort, Dual Pivot Quicksort.

## 1 Introduction

Since their introduction, software systems have become increasingly complex, rendering verification and maintenance more difficult and time consuming. As a consequence, it is becoming more and more difficult to identify possible bug threats using traditional techniques. In the modern software engineering industry, formal verification methods are necessary in order to ensure the correctness of widely used software.

A particular class of widely used software is represented by sorting algorithms. The main aspects of sorting algorithms that researchers have studied over the years are running time complexity and correctness. In this paper, we focus our analysis on the latter.

While in the early days of computer science people used handwritten proofs for algorithms, this has become a thing of the past with the emergence of specialized software tools, called theorem provers.

Theorem provers use the principles of Automated Theorem Proving (ATP)[1] to verify the correctness of software programs. They use first-order logic (also known as predicate logic) or higher-order logic to represent the specifications (hypothesis and axioms) of a problem and show that some statement (the conjecture) is the logical consequence of these specifications.

There are several theorem provers that have been successfully used in the field of software verification. In the particular case of sorting algorithms, a proof of Mergesort was formalized using the theorem prover Isabelle[2], while formal proofs for Insertion sort, Heapsort and Quicksort were achieved by using the proof assistant Coq[3].

In this paper, we will analyze the use of another theorem prover, called KeY[4], in the process of formally verifying 4 sorting algorithms: Counting Sort, Radix Sort, TimSort, and Dual Pivot Quicksort. We will also provide an overview of these algorithms, highlighting their working principles, as well as a description of the verification tool used.

## 2 KeY: the verification tool

KeY is a system that was developed in a joint effort by researchers from the University of Karlsruhe, Chalmers University of Technology in Göteborg and the University of Koblenz. An overview of this verification tool, as well as a non-trivial use case are presented in the paper *Verifying Object-Oriented Programs with KeY: A Tutorial*[5].

KeY supports two main specification languages: Object Constraint Language (OCL)[6], mostly used by people familiar with UML, and Java Modeling Language(JML)[7], popular with the Java programmers. Java Card DL (Dynamic Logic) is the target language used for verification. Internally, KeY uses a *taclet* language for the description of rules. The taclets specify the logical content and the context of a rule. A deductive verification component is at the core of the KeY system and it uses a free-variable sequent calculus for first-order Dynamic Logic for Java.

At the root of KeY lies a typed first-order predicate logic with subtyping. It also uses parameterized modal operators $\langle p \rangle$ and $[p]$, where $p$ can be any sequence of Java Card statements. Other features of the KeY logic are type casts (changing the static type of a term) and type predicates (checking the dynamic type of a term), which allow it to analyze OOP features of Java programs, such as inheritance and polymorphism. The type hierarchy contains the primitive data types (e.g. integer, boolean), the type Null, the root type Object, and also user-defined types, representing Java classes and interfaces. It also includes the infinite integer type $\mathbb{Z}$, in addition to the range-limited integer types of Java. Along with predefined symbols, KeY also supports user-defined predicates and functions, which can be rigid (i.e. they have the same meaning in all program states) or non-rigid (i.e. they can differ from state to state). Another important feature are the *updates*, which are model operators that describe program transitions and correspond to assignment operations in imperative languages.

When working with KeY, the user can switch between an automated proof-search mode and an interactive mode. For the latter, a graphical user interface is used, which allows the manual application of rules. All rules are defined in the so called ”taclet language”, which has a clear and easy to understand syntax, illustrated by the example below.

```
\find (p -> q ==>) //implication in antecedent
\assumes (p ==>)        //side condition
\replacewith(q ==>)     //action on focus
\heuristics(simplify)  //strategy information
```

Thanks to its easy to use graphical user interface and clear syntax, the KeY system has been used to teach logic, deduction and formal methods to computer science students. However, its adoption has not been limited to the academic environment. Several other applications include the verification of a Java implementation of the Schorr-Waite graph marking algorithm[8], used in JVM garbage collectors, the verification of a module of a flight management system designed by Thales avionics[9], and the verification of Java implementations of several sorting algorithms. The latter will be presented in the following section.

## 3 Sorting algorithms

In this section we will provide a brief overview of each algorithm, their basic idea and how they are implemented. We will then describe the verification processes presented in their respective papers [10] [11] [12].

- *Dimitris Laskaratos is an MSc. Computing Science student at the University of Groningen, E-mail: d.laskaratos@student.rug.nl.*
- *Bogdan Petre is an MSc. Computing Science student at the University of Groningen, E-mail: b.petre@student.rug.nl.*

### 3.1 Counting Sort

As opposed to most sorting algorithms, Counting Sort does not depend on comparisons between the objects, but takes into account the number of occurrences of each object in the array that is to be sorted. It then uses addition and subtraction to calculate their position in the output sorted array.

#### 3.1.1 Overview

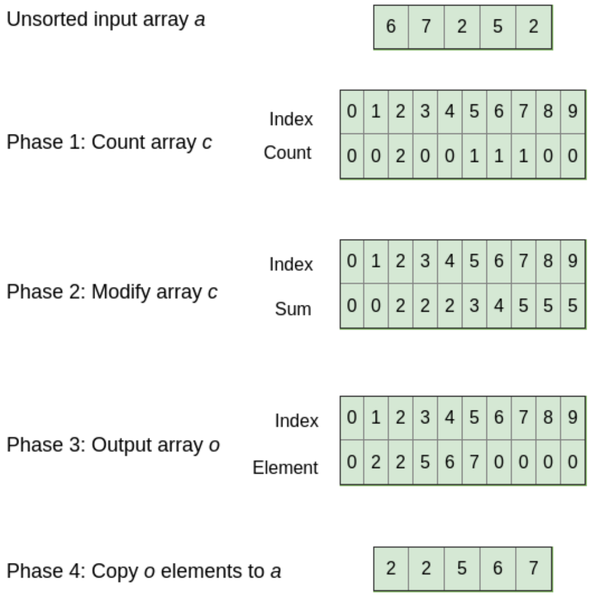
Counting Sort works as follows: consider an input array *a* containing random non-negative integers in a certain range, for example 0 to 9, and a second array *c* that stores the number of occurrences of each number in the range. The first step is then to fill array *c*. Intuitively, the size of this array is the same as the range of the input, that is, 10. The contents of the array would be 0 if the number 0 is not in the input, 2 if number 1 occurs twice and so on. The second step is to modify the *c* array so that each element is the sum of its value and the value of the preceding element. The next step is to store the elements of the input array to a new array, let that be *o*. Only this time, their index will be their count from array *c*, followed by a decrease by 1. So if element 1 occurred twice in the input, its index in array *a* will be 2, with its new count now being 1. In the end, array *o* will contain the sorted input elements. An illustration is available in pseudo-code in Listing 1 and in Figure 1.

Listing 1: Pseudocode for Counting Sort

```
Initialize input array 'a';
Initialize count array 'c';
Initialize output array 'o';

for each element of 'a':
    count its occurrences in 'a';
    store the count in
    the corresponding index of 'c';
end

for each element of 'c':
    calculate the sum of the
    current and previous values;
    store the sum in the current position;
end

for each element of 'a':
    store element in output
    array at position count[element];
    decrease count[element] by 1;
end

Copy the first size(a) elements of 'o'
starting from position 1, to 'a';
```

General remarks about Counting Sort:

- It can work with any range.
- Worst case time complexity is $\mathcal{O}(n+k)$, where n is the number of elements in 'a' and k is the range (boundary) of the input elements.
- The algorithm uses partial hashing to count element occurrences in O(1) time.
- Counting Sort works well with non-negative integers but can also be extended to work with negative inputs.
- Position 0 of the output array will always remain empty.



Unsorted input array *a*

|   |   |   |   |   |
|---|---|---|---|---|
| 6 | 7 | 2 | 5 | 2 |

Phase 1: Count array *c*

| Index | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
|---|---|---|---|---|---|---|---|---|---|---|
| Count | 0 | 0 | 2 | 0 | 0 | 1 | 1 | 1 | 0 | 0 |

Phase 2: Modify array *c*

| Index | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
|---|---|---|---|---|---|---|---|---|---|---|
| Sum | 0 | 0 | 2 | 2 | 2 | 3 | 4 | 5 | 5 | 5 |

Phase 3: Output array *o*

| Index | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
|---|---|---|---|---|---|---|---|---|---|---|
| Element | 0 | 2 | 2 | 5 | 6 | 7 | 0 | 0 | 0 | 0 |

Phase 4: Copy *o* elements to *a*

|   |   |   |   |   |
|---|---|---|---|---|
| 2 | 2 | 5 | 6 | 7 |

Figure 1: Counting Sort example

#### 3.1.2 Verification

The verification process of Counting Sort, as described by S. de Gouw et al.[10], focuses on proving the correctness and stability[1] of the algorithm using the KeY tool. Several assumptions were made to ensure a more comprehensive demonstration, such as that the arrays consist of small integers, they are always allocated and are unbounded (they are dynamically allocated). The second assumption may be redundant though, as Java always initializes integer arrays with zeros.

The first step is to define a contract in JML that specifies the correctness and stability of a generic sorting algorithm. Practically, the purpose of this step is to declare a set of rules (or constraints) that dictate how a correct and stable algorithm should behave, with respect to the resulting array. These rules are essential because they are the basis for the proof process that follows. The contract itself is not important to this paper, so we will only mention the rules applied in [10]. These are:

- A precondition *requires*, setting the boundary for the integers in the array.
- A postcondition *ensures*, stating that the resulting array should be a permutation of the input array, and also that it is to be sorted up to a specified index.
- A predicate expression in the postcondition which evaluates to true if there is a bijection[2] from the input array to the resulting array.

Next, in order for the proof to be more easily understandable, some auxiliary functions are defined, which can be seen in Table 1. These functions represent the result of operations conducted by the algorithm during the sorting, in the JML context. For instance, the *val* function stores the number represented in base *b*, which in turn is stored in row *r* of the large numbers array *a*. This number has *d*+1 digits.

Lastly, these functions are used to prove that Counting Sort complies with the specifications given by the JML contract. Specifically, the authors of [10] use them in invariant loops written in JML format,

[1] Stability is the property that objects with the same key appear in the same order in the sorted array as in the input array.

[2] Bijection is a property in which each element of the resulting array corresponds to exactly one element of the input array

| Function | Operation |
|---|---|
| val(b,r,d,a) | $\sum_{i=0}^{d}(a[r][i] * b^i)$ |
| cntEq(x,r,a) | $\lvert\{i \mid 0 \le i \le r \wedge a[i] = x\}\rvert$ |
| cntLt(x,r,a) | $\lvert\{i \mid 0 \le i \le r \wedge a[i] < x\}\rvert$ |
| pos(x,e,l,a) | cntEq(x,e,a) + cntLt(x,l,a) |

Table 1: Auxiliary functions

representing the loops explained in the pseudo-code. Subsequently, through a series of equalities, they prove that each of the rules mentioned above are satisfied and Counting Sort is thus correct.

## 3.2 Radix Sort

Radix Sort is a non-comparative integer sorting algorithm, used for sorting arrays of large numbers digit by digit, making use of another, implementation-defined, stable sorting algorithm.

### 3.2.1 Overview

We provide the pseudocode of Radix Sort in Listing 2. Note that "LSD" and "MSD" stand for least significant digit and most significant digit. An example array, sorted using Radix Sort, is given in Figure 2.

Listing 2: Pseudocode for Radix Sort

```
Input: L = list of unsorted integers
Output: Sorted list L
for i between LSD and MSD:
        stable sort L based on the i-th digit
```

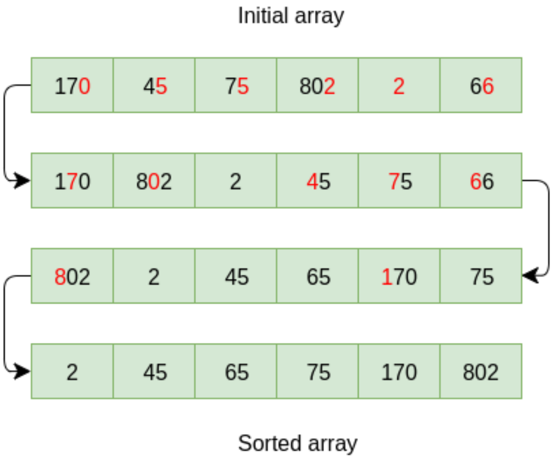


Figure 2: Radix Sort example

The time complexity is linear in the time complexity of the stable sorting algorithm used. One of the preferred stable sorting algorithms used in the implementation of Radix Sort is Counting Sort. This is because the time complexity of Counting Sort is linear, namely $\mathcal{O}(n+k)$, as mentioned in the previous section, where $n$ is the number of elements is the list and $k$ is the range of elements, which in the case of Radix Sort is the base of representing the numbers (e.g. $k = 10$ for the decimal system).

### 3.2.2 Verification

A formal proof of a Java implementation of the Radix Sort algorithm is presented by Stijn de Gouw, Frank de Boer and Jurriaan Rot in [10]. As is to be expected, the correctness of Radix Sort relies on the correctness of the stable sorting algorithm used. The authors, in their proof of Radix Sort, assume a generic stable sorting algorithm is used. This has the advantage of extending the proof from just the Radix Sort implementation using Counting Sort, to implementations that use any other stable sorting algorithms.

The issue with this more general approach was that the stable sorting JML contract described in Section 3.1.2 is only valid for arrays of integers, while the Java implementation of Radix Sort operates on arrays of large numbers, not necessarily integers. The authors proceeded by adapting the JML contract to work with large numbers. Following this adjustment, they proved the correctness of Radix Sort.

## 3.3 TimSort

TimSort is a stable sorting algorithm, that was first implemented in 2002 by Tim Peters in Python. It is currently both the default sorting algorithm used in Python since version 2.3 and also the default implementation of **java.util.Arrays.sort()** and **java.util.Collection.sort()** of the OpenJDK library.

### 3.3.1 Overview

TimSort is a relatively complicated sorting algorithm, that uses different sorting strategies depending on the input. It uses concepts and techniques presented by Peter McIlroy in his 1993 paper "Optimistic Sorting and Information Theoretic Complexity"[13]. McIlroy started from the observation that, although most permutations will require $\mathcal{O}(n\log n)$ comparisons, we should identify classes of permutations for which fewer comparisons are necessary and use them to design sorting algorithms. He then presented several variations of Insertion Sort and Merge Sort and analyzed their performance over different classes of incompletely shuffled arrays.

TimSort takes advantage of a desirable property of unsorted lists, described by McIlroy in [13]: *natural runs*. These are continuous sequences of maximum length that are either ascending[3] or strictly descending[4] (Figure 3).

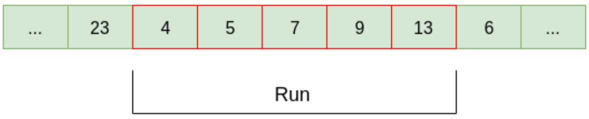


Figure 3: Run example

The algorithm starts by searching for runs. If it finds a run that is smaller than the minimum run size (*minrun*), it uses binary insertion sort to increase the size of the run to *minrun*. It then pushes the run into a stack of runs. The value of *minrun* is selected so that the number of runs becomes a power of 2 or slightly less, because that is when merge sort has the best performance.

After every insertion of a new run in the stack of runs, the algorithm reexamines the stack and uses merge sort to combine adjacent runs until the invariants in Listing 3 are reestablished. C, D and E represent the 3 runs at the top of the stack in the order of addition (E is at the top), while $|X|$ represents the length of run $X$.

Listing 3: Invariants of TimSort

```
1. |C| > |D| + |E|
2. |D| > |E|
```

If the first invariant is broken, the algorithm merges runs C and D. If the second invariant is broken, runs D and E will be merged, as shown in Figure 4.

The two invariants are required for the purpose of having balanced merges, a situation in which Merge Sort performs best. To preserve the stability property, only adjacent runs can be merged. For the actual merge of 2 runs, a Merge Sort with binary search is used, as described by McIlroy in [13], under the term "Merge Sort with Exponential Search".

[3] Each element is greater or equal than the previous.

[4] Each element is less than the previous. These runs are then reversed and they need to be *strictly* descending in order to preserve stability.

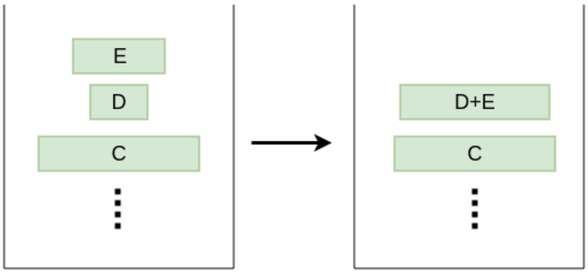


Figure 4: Reestablishing the second invariant

TimSort is called an adaptive sorting algorithm because it takes advantage of existing order in its input (i.e. runs). It is also a hybrid sorting algorithm because it uses both Merge Sort and Binary Insertion Sort. The worst and average case time complexities of TimSort are $\mathcal{O}(n \log n)$, while the best case is $\mathcal{O}(n)$, when the input is already sorted.

### 3.3.2 Verification

In 2015, Stijn de Gouw et al. analyzed the Java implementation of TimSort, used by OpenJDK as the default sorting algorithm for arrays and collections [11]. They found that for some run lengths, the invariants do not hold. Such an example is illustrated in Figure 5, showing the stack containing runs A, B, C, D, E, with their respective lengths in parentheses, right after run E was pushed in the stack. Note that, up to that point, the invariants held for runs A, B, C, D. Trying to reestablish the invariants, the algorithm merges C and D. Although the invariants now hold for B, C+D, E, the first invariant does not hold for A, B and C+D because $120 < 80 + 45$.

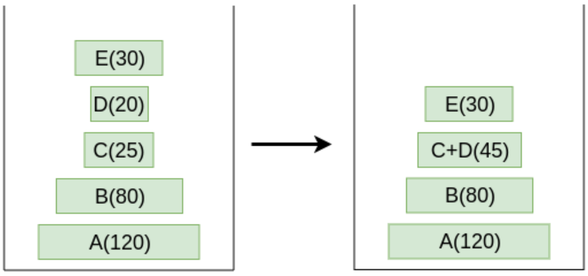


Figure 5: Breaking the invariant

The issue is that the length of the array that holds the lengths of the runs, **runLen**, is chosen based on the length of the input array and on the assumption that the invariants are always respected. When designing the algorithm, the goal was to choose the size of **runLen** small enough for the algorithm to be fast, but larger than the size of the stack holding the runs, **stackSize**. In [11] it is showed that certain worst-case input arrays can cause the latter condition to fail. Table 2 shows that for large input arrays and particular inputs, the required stack size is larger than the size of **runLen**, which results in a ArrayIndexOutOfBoundsException, breaking the algorithm.

In [11], the authors also propose an updated implementation of TimSort, in which the invariants are always satisfied. This was achieved by performing invariant checks on the top 4 elements of the stack, instead of just the top 3, as in the original version. To verify this new implementation, KeY was used.

Similar to the verification procedure used to prove Counting Sort correct, 4 auxiliary predicates, related to the invariants, are introduced to obtain a human readable specification. These are presented in Table 3, the last being defined as the logical conjunction of the first 3.

Using these 4 abstractions, the next step was to write the JML contracts in KeY for each of the 3 main parts of the algorithm: the main loop of TimSort, the *pushRun* function, which pushes runs into the stack, and the *mergeCollapse* function, which uses merge sort to merge 2 runs. Having done all these steps, KeY showed that the proposed algorithm was indeed correct.

| array size | required stack size | runLen.length |
|---|---|---|
| 64 | 3 | 5 |
| 128 | 4 | 10 |
| 160 | 5 | 10 |
| 65536 | 21 | 24 |
| 131072 | 23 | 40 |
| 67108864 | 41 | 40 |
| 1073741824 | 49 | 40 |

Table 2: Worst-case stack sizes

| **Name** | **Definition** |
|---|---|
| elemBiggerThanNext2(arr, idx) | $(0 \leq idx \wedge idx+2 < arr.length) \rightarrow arr[idx] > arr[idx+1] + arr[idx+2]$ |
| elemBiggerThanNext(arr, idx) | $(0 \leq idx \wedge idx+1 < arr.length) \rightarrow arr[idx] > arr[idx+1]$ |
| elemLargerThanBound(arr, idx, v) | $(0 \leq idx < arr.length) \rightarrow arr[idx] > v$ |
| elemInv(arr, idx, v) | $elemBiggerThanNext2(arr, idx) \wedge elemBiggerThanNext(arr, idx) \wedge elemLargerThanBound(arr, idx, v)$ |

Table 3: Auxiliary predicates

## 3.4 Quicksort

QuickSort, much like MergeSort, uses a 'divide and conquer' strategy by selecting an element from the array as pivot. It then separates the given array around this pivot and recursively partitions the resulting arrays. QuickSort has several variations that differ in the way they select the pivot (first element, last element, middle element, random element). Beckert et al. [12] attempted to prove the correctness of a dual pivot QuickSort, which is the matter of discussion in this section. Below, a general overview of single pivot QuickSort is given, followed by a brief explanation of the dual pivot version.

### 3.4.1 Overview

QuickSort can work with any input, but for simplicity let us assume that the input array contains small integers. The first step of the algorithm is to select an element from the array to serve as pivot, by one of the methods mentioned above. For the purposes of this example, let pivot be the last element in the sequence. The input array is then partitioned around this element using the following process: the array is reordered so that all elements smaller than the pivot are positioned in its left side, while elements with bigger value are positioned on its right side. The pivot element is now in its final position and QuickSort can now be applied to the resulting arrays left and right of it.

Intuitively, the most important aspect of the algorithm is the partition process. In an attempt to explain this operation in depth, we have provided an illustration in Listing 4 and in Figure 6. Besides the pivot, two additional pointers are kept, that point at the leftmost and rightmost elements before the pivot respectively. At first, control loops from left to right until it finds an element larger than the pivot. Then it loops from the rightmost element before the pivot to the left until it finds an element smaller than the pivot. If $leftElement > rightElement$, they are swapped and the loop starts again at the current pointers' position. The operation ends when the leftmost element larger than the pivot is swapped with the pivot. The array now contains all elements smaller than the pivot on the left side of the pivot, and all larger elements on the right side. The same operation then begins for each of the two arrays. When all sub-arrays have been sorted, they are concatenated again to form the output sorted array.

Listing 4: Partition operation of QuickSort

```
function partition(left, right, pivot)
    leftpointer=left -1;
              rightpointer=right;

    while True
        while array[leftpointer]<pivot do
            leftpointer++;
        end while

        while array[rightpointer]>pivot do
            leftpointer --;
        end while

        if leftpointer >=rightpointer
            break;
        else
            swap leftpointer ,rightpointer;
        end if

    end while

    swap leftpointer , right;
    return leftpointer;

end function
```

General remarks about QuickSort, with $n$ being the number of elements:

- Worst case time complexity is $\mathcal{O}(n^2)$.
- Average case complexity is $\mathcal{O}(n\log n)$.
- The algorithm is efficient for large-sized data.
- If implemented efficiently, it can be a lot faster than MergeSort and HeapSort.

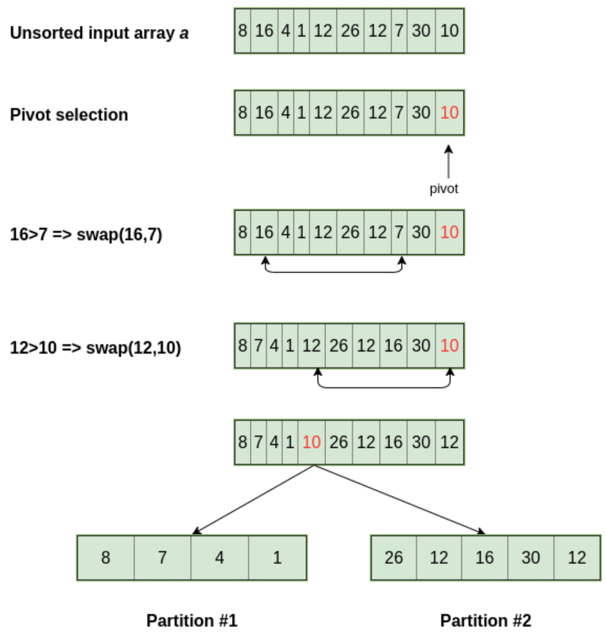


Figure 6: Partition operation in QuickSort

As mentioned, Beckert at al. [12] work with a version of QuickSort that uses two pivots, proposed by Vladimir Yaroslavskyi in 2009. While in single pivot QuickSort, the pivot splits the input array in two, in dual-pivot QuickSort, the two pivots, p1 and p2, separate the array in three sections, by comparing the elements against both pivots. The first section on the left side contains the elements that are smaller than the smaller pivot, the middle section contains the elements with values between the two pivots, which is also inclusive, and the last section on the right side contains the elements that are larger than the pivots. All three parts are then recursively sorted by the same process discussed previously. Research showed that dual pivot QuickSort performed better than the single pivot version, which led to its adoption in OpenJDK 7, becoming the standard sorting algorithm for primitive data.

### 3.4.2 Verification

As with the aforementioned algorithms, for the verification of QuickSort, the KeY tool was used and, by extension, JML. The JML contract specified the following rules that needed to be satisfied: the sorted array should be in increasing order, the sorted array should be a permutation of the input array, should not modify any memory other than the one used for the input array, the method must terminate and should not throw an exception.

As the implementation of QuickSort in JDK is rather large, Beckert at al. decided to break the algorithm into smaller properties and prove their correctness individually. These properties are ”sortedness” and permutation.

The sorting property of the algorithm was the easiest to prove, since it merely requires arithmetic comparisons between the elements of the output array. A relatively simple JML snippet was used to assert the correctness of this property.

The permutation property of QuickSort is also very easy to prove since there is a continuous swapping of the elements and as such, the resulting array is a permutation of the original. However, in the Java implementation, the two pivots are excluded during the recursive calls, which means there is no proof that after their restoration the array is a permutation of the original one. The writers used KeY to design a lemma involving a mapping factor $\sigma$ such that, $b[i] = a[\sigma(i)]$, for all $0 \leq i < n$, ultimately proving that both pivots are present in the intermediary array.

After proving the above properties, the proof of correctness for the QuickSort algorithm was completed successfully. It was noted that the process took roughly two months to conclude.

## 4 Discussion/Results

KeY was used to prove the correctness of the algorithms. It provided the means to prove the theorems and lemmas constructed by the authors, to conduct a formal verification of Counting Sort, RadiX Sort, Dual Pivot QuickSort and an updated implementation of TimSort. What was evident in the reports, is the fact that in order to formally prove the correctness, a lot of rules had to be applied, specifying a behavior that is taken for granted. For example, satisfying the permutation requirement in QuickSort was not as simple as other requirements, because the code implementation treated the pivots in an unexpected way, which in turn, called for additional application of rules.

The basic idea that was followed to verify the aforementioned algorithms was pretty much common among them, as they were the requirements: the resulting array has to be sorted and has to be a permutation of the input array. For Counting Sort and Radix Sort the JML code to test the corresponding rules was straightforward and understandable, but for QuickSort it took many more scripts and rules, probably due to its higher complexity compared to the others.

## 5 Conclusions

The following conclusions are derived after our investigation was concluded:

- As was to be expected, a formal verification procedure requires mathematical precision and takes into account aspects of the implementation code and the algorithmic logic that are considered trivial by most developers. There are a number of tools and methods with which to conduct such a procedure, with the most Java-oriented of them being KeY.

- In order to verify mainstream complex algorithms, it is essential to break down the problem into smaller tasks, and prove each of them individually.
- Formal verification methods detected a bug in TimSort that would probably not reveal itself otherwise.
- In the case of complex algorithms, formal verification may take a long time to complete, as in the case of the two-month period required for the verification of QuickSort.

Following the success of KeY in verifying these sorting algorithms, the methods showcased in this paper can be used for the verification of other algorithms used in JDK, such as the implementation of SHA-256 (for hashing), Binary Search or the algorithms used for regular expressions.

## Acknowledgements

We wish to thank the expert reviewer, professor Gerard Renardel, as well as our colleagues, Johan de Jager and Thijs Klooster, for providing valuable feedback for this paper.

## References

[1] Frank Pfenning. *Automated Theorem Proving*. Carnegie Mellon University, first edition, 2004.

[2] Christian Sternagel. Proof pearl—a mechanized proof of ghc's mergesort. *Journal of Automated Reasoning*, 51(4):357–370, Dec 2013.

[3] Jean-Christophe Filliâtre and Nicolas Magaud. Certification of sorting algorithms in the coq system. 1999.

[4] Bernhard Beckert, Reiner Hähnle, and Peter H. Schmitt. *Verification of Object-oriented Software: The KeY Approach*. Springer-Verlag, Berlin, Heidelberg, 2007.

[5] Wolfgang Ahrendt, Bernhard Beckert, Reiner Hähnle, Philipp Rümmer, and Peter H. Schmitt. Verifying object-oriented programs with key: Atutorial. In Frank S. de Boer, Marcello M. Bonsangue, Susanne Graf, and Willem-Paul de Roever, editors, *Formal Methods for Components and Objects*, pages 70–101, Berlin, Heidelberg, 2007. Springer Berlin Heidelberg.

[6] Jordi Cabot and Martin Gogolla. *Object Constraint Language (OCL): A Definitive Guide*, pages 58–90. Springer Berlin Heidelberg, Berlin, Heidelberg, 2012.

[7] Lilian Burdy, Yoonsik Cheon, David R. Cok, Michael D. Ernst, Joseph R. Kiniry, Gary T. Leavens, K. Rustan M. Leino, and Erik Poll. An overview of jml tools and applications. *International Journal on Software Tools for Technology Transfer*, 7(3):212–232, Jun 2005.

[8] Richard Bubel. *The Schorr-Waite-Algorithm*, pages 569–587. Springer Berlin Heidelberg, Berlin, Heidelberg, 2007.

[9] Peter Schmitt, Isabel Tonin, Claus Wonnemann, Eric Jenn, Stéphane Leriche, and James J. Hunt. A case study of specification and verification using jml in an avionics application. In *Proceedings of the 4th International Workshop on Java Technologies for Real-time and Embedded Systems*, JTRES '06, pages 107–116, New York, NY, USA, 2006. ACM.

[10] Stijn Gouw, Frank Boer, and Jurriaan Rot. Proof pearl: The key to correct and stable sorting. *J. Autom. Reason.*, 53(2):129–139, August 2014.

[11] Stijn de Gouw, Jurriaan Rot, Frank S. de Boer, Richard Bubel, and Reiner Hähnle. Openjdk's java.utils.collection.sort() is broken: The good, the bad and the worst case. In Daniel Kroening and Corina S. Păsăreanu, editors, *Computer Aided Verification*, pages 273–289, Cham, 2015. Springer International Publishing.

[12] Bernhard Beckert, Jonas Schiffl, Peter H. Schmitt, and Mattias Ulbrich. Proving jdk's dual pivot quicksort correct. In Andrei Paskevich and Thomas Wies, editors, *Verified Software. Theories, Tools, and Experiments*, pages 35–48, Cham, 2017. Springer International Publishing.

[13] Peter McIlroy. Optimistic sorting and information theoretic complexity. In *Proceedings of the Fourth Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '93, pages 467–474, Philadelphia, PA, USA, 1993. Society for Industrial and Applied Mathematics.

# Audio Event Detection: Current State and Future Developments

Tim Oosterhuis, Daan Opheikens

**Abstract**— Audio event detection and classification are becoming more and more important for different areas in science. However, audio event detection and classification still pose a challenge in real world environments with non-stationary background noise. Multiple different approaches to solve this problem are outlined in this paper, both for classification of isolated sound events (referred to in this paper as AEC), and for detection and subsequently classification of sound events in a larger audio stream (henceforth AED). AEC techniques include support vector machines, hidden Markov models, K nearest neighbor (KNN) classifiers, deep neural networks and convolutional neural networks (CNN's). Research on the best audio features to use is still ongoing. Different AEC techniques have been tested on a benchmark data set taken from the Real Word Computing Partnership (RWCP) Sound Scene Database in Real Acoustic Environments. On this data set, a 1 layer CNN had the highest mean classification accuracy at 98.6%, and a relatively simple KNN classifier trained on the sub-band power distribution image features (SPD-IF's) of sounds was the second most successful at 96.0%. With regard to AED, techniques based on random regression forests and trainable sound filters seem promising, but lack of benchmark data set standardization prevents quantitative comparison between these solutions.

**Index Terms**—Audio event detection, audio event classification, feature extraction, representation learning

## 1 Introduction

Dynamic audio event detection is the automatic recognition of sounds in unconstrained potentially noisy environments. For example, outside environments or loud inside environments, such as busy restaurants and public transports. The goal is to detect and correctly identify an event by its signature sound in a stream of noisy audio. For example: a conversation, an alarm, or the tire slip of a car.

The detection and classification of audio events has various applications in areas such as auditory surveillance of public roads or bathrooms, multimedia integration, machine hearing in robotics, music recognition and automated meeting annotation.

For many of these applications the environment may contain non-stationary background noise, which makes audio event detection and classification challenging, compared to speech recognition. Furthermore, for some applications, particularly in robotics and surveillance, failure to detect a relevant audio event in time could potentially have dramatic consequences.

Due to background noise, methods traditionally applied in speech recognition often fail. However, a number of different machine learning techniques and novel sound features have been explored recently, which have reportedly been successful under noisy conditions. Advances have been made both in classifying isolated sound events corrupted by noise (audio event classification) and in detecting sound events within a larger audio stream with background noise (audio event detection). See also figure 1 for a graphical depiction. Audio event detection and classification are closely related. Detection is usually followed by classification, and can be done with a binary event/background classifier, in which case it is called detection-by-classification in literature [14].

In order to make a clear distinction between methods which only focus on the classification of 'pre-detected' isolated sound events and methods which include the detection of an audio event as well as the subsequent classification of the detected sound event, the former will be be referred to with the acronym AEC in this paper and the latter will be referred to with the acronym AED.

In section 2, we will describe the different sound features and machine learning architectures which have recently been used in audio event detection and classification. The set up of our quantitative evalu-

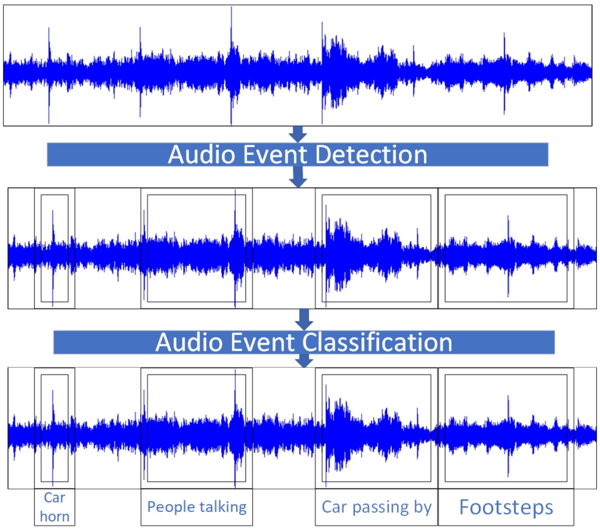


Fig. 1. Graphic highlighting the concepts of audio event detection and classification as used in this paper.

- *Tim Oosterhuis is with Rijksuniversiteit Groningen, E-mail: tim.oosterhuis@hotmail.com.*
- *Daan Opheikens is with Rijksuniversiteit Groningen, E-mail: daan_opheikens@hotmail.com.*

ation using the results from recent literature will be made in section 3.1 The results of this evaluation will be presented in section 3.2. The implications of these results and potential future work will be discussed in section 4. Finally, we will conclude our research into the state-of-the-art of audio event detection and classification in section 5.

## 2 State of the art

### 2.1 Audio signal features

The detection and recording of audio signal features can be done in various ways. Usually, the parameters are extracted on a frame-by-frame basis. These are also known as short-time features. Jimmy Ludea-Choez and Ascensin Gallardo-Antoln [7] give several examples of different kinds of techniques used in the detection of audio features; the conventional Mel-Frequency Cepstral Coefficients (MFCC), log filter bank energies, Perceptual Linear Prediction (PLP), log-energy, spectral flux, entropy and zero-crossing rate [9].

However, these techniques are all designed for speech, which is quite different from other audio events. Certain audio events might have temporal qualities. For example: the ringing of a telephone or a piece of music. To make use of these qualities, the techniques used for speech detection are falling short. Therefore, new audio detection techniques are being developed; new acoustic parameters such as Power Normalised Cepstral Coefficients (PNCC) [4] and those derived from Gammatone [6] or Gammachirp filter banks [1] can deal with the difference between speech and other audio events. Other techniques are the application of Non-Negative Matrix Factorization (NMF) or K-Singular Value Decomposition (KSVD) on audio spectrograms [11].

By using spectro-temporal features (combining temporal and spectral analysis), the problem that appears with temporal qualities can be lessened, as proven by Jimmy Ludea-Choez and Ascensin Gallardo-Antoln[9]. The downside of this method, is that it is very demanding in terms of computational resources. An alternative to using spectro-temporal features is combining the short-time parameters that appear in long audio segments, as this requires less computational power. It allows us to see the dynamic structure in short-time features, which can make it easier to detect a pattern or other temporal features.

Jimmy Ludea-Choez and Ascensin Gallardo-Antoln propose a new technique they call 'Short-time feature extraction based on spectral band selection'. The objective of their approach is to create subsets of features that are useful for classification. The idea behind this technique is "to take advantage of the specific spectral and temporal patterns of acoustic events for enhancing the representation and discrimination capabilities of the extracted features."[9] An advantage of short-time feature extraction based on spectral band selection is that it is modular, which helps in computational power. The disadvantage of this technique is that properties of the human hearing, such as temporal and frequency masking, are not taken into account. This might give the detector data that, although correct, is not useful for the classifier to work with.

Huy Phan [14] uses the detection-by-classification scheme technique for AED, which extracts global representations for isolated events in training data. After this, classification models are trained to distinguish the events from background as well as classify them into different classes. After the detection of the audio, the data is passed on to the classifiers.

### 2.2 Classification architectures

#### 2.2.1 Feature extraction based architectures

There have been a number of different machine learning architectures, which have been applied to audio event classification. These methods include methods based on hidden Markov models (HMM) and support vector machines (SVM) as well as a deep neural network (DNN) and convolutional neural networks (CNN). CNN's will be discussed in section 2.2.2.

Dennis et al. [3] used a K nearest neighbor classifier in combination with their sub-band power distribution and spectrogram image features (SPD-IF). They experimented with both euclidean and Hellinger distance and found the Hellinger distance based K nearest neighbor to be the most successful when applied to the SPD-IF.

McLoughlin et al. [10] used a deep neural network architecture, which they trained on features extracted from the auditory image and spectrogram image (referred to as AIF and SIF respectively, see section 2.1 for a more complete description of the features used) of noisy sound signals. They experimented with different numbers of hidden layers and different hidden units per layer. They reported their best result (in terms of a trade off between validation accuracy and computational resources) with networks consisting of two hidden layers with around 200 hidden units in each layer.

Strisciuglio et al. [15] have developed a audio event classifier which filters sound events from a stream sound data using a Combination Of Peaks of Energy (COPE). This method is biologically inspired by the way sound signals are transformed into neuron firing responses in the human ear. The filtered sound frames are classified using a multi-class SVM. Prior to training, the COPE filter for each class is configured based on a class prototype.

#### 2.2.2 Convolution based architectures

Convolutional neural networks are a powerful deep learning architecture, which has proven to be successful at classification problems with image input, including speech frequency spectrograms. A CNN typically consists of multiple convolutional layers starting from the input layer which are connected through each other via randomly initialized convolution filters which act as weights. Leading up to the output layer is one or more fully connected layer. See [8] for a more detailed description of the CNN architecture. In contrast to feature based learning architectures, CNN's are trained directly on raw input data, requiring minimal manual preprocessing. Due to the CNN architecture lower level representative features of an input image or signal are stored in the lower convolutional layers of the network.

Zhang et al. [16] and Phan et al. [13] have both trained an CNN classifier on the same data set taken from the RWCP and previously used by [3] and [10]. Zhang et al. used a network with two convolutional layers each followed by a subsampling layer, whereas Phan et al. used a network with one convolutional layer followed by a max-pooling layer.

In contrast to [16] and [13] where labeled sound data was used for training, Aytar et al. trained a one-dimensional CNN on the sounds from unlabeled video [2]. A trained object and scene recognition CNN was used to train the sound recognition network, using the results from the visual frames to evaluate the audio frames. This experimental setup was created to overcome the problem of scarcity in large labeled dynamic sound data sets.

Aytar et al. use raw sound wave data as input to their audio event classifiers, whereas Zhang et al. and Phan et al (2016) use a two-dimensional time-frequency representation of the sound wave data as input. Zhang et al. have experimented with smoothing their input spectrogram.

### 2.3 Audio event detection pipelines

The precise detection of an sound event in a larger audio stream, including the estimation of said sound event is usually handled as a two step classification problem [14]. The first step is the separation of foreground and background sound, which is typically done on all frames of a large audio file in a sliding window fashion. The second step in this case is the multi-class classification of all detected foreground sounds. In contrast to this two step detection-by-classification approach, Phan et al. (2015) [14] propose a three step AED process. After separation of background sound frames, and classification of sound event frames, they added a last step where they estimated the temporal onset and offset of the classified sound events in time using regression forests. This allowed them to more accurately localize detected sound events in time.

## 3 Evaluation

### 3.1 Setup

The performance of the different AEC architectures and features discussed in section 2 will be evaluated quantitatively, using the classification accuracy reported in the various papers in which the architectures and features are proposed.

#### 3.1.1 Data sets

Several of the different AEC architectures have been evaluated on a data set, taken from the Real Word Computing Partnership (RWCP) Sound Scene Database in Real Acoustic Environments. This allows for a straightforward quantitative comparison of performance within this group of architectures. The group of papers in which the RWCP data set is used, which are reviewed in this paper are [3], [10], [16] and [13]. The RWCP data set consisted of 50 different sound event (SE) classes , with 80 sound files per class. The test data of the RWCP group included clean SE data, as well as noisy SE data reduced by 20db, 10db, and 0db; The 0db reduction condition being the most challenging. The noisy SE data was generated by mixing the RWCP data with background noises taken from the NOISEX92 database, including ‘factory floor’, ‘speech babble’, ‘destroyer control room’ and ‘jet cockpit 1’ noise. This was done to simulate real life non-stationary noise conditions. Another data set which has been used is based on the MIVIA roads event data base. The MIVIA roads event data set was used by Striscuglio et al. [15] who included earlier results by Foggia et al. [5] as a baseline. Sound events from the MIVIA data base (car crashes and tire skids) with ‘normal’ road sounds to form non-stationary background noise. The data set included 849 seconds of sound events and 2732 seconds of background noise. Lastly, Phan et al. (2015) [14] evaluated their solution with the ITC-IRST and UPC-TALP databases and included baseline results based on SVM and HMM classifiers. The ITC-IRST database and UPC-TALP data both consist of conference meeting audio streams with 767 sound events divided over 17 classes for ITC-IRST and 1028 sound events divided over 14 classes for UPC-TALP. Phan et al. split these audio streams into several hundred thousands of densely overleaved superframes per data set, with the ratio of event versus background superframes being approximately three to one for both sets (see [14] for the exact number of superframes per sound event class).

#### 3.1.2 Performance metrics

The group of papers for which the RWCP data set is used which are reviewed in this paper are focused purely on correct classification of isolated audio events. The performance metric used by the authors of these papers to report their results is the classification accuracy, which is the measure of the correctly classified sound events as a percentage of the total amount of sound events in the test set [3] [10] [16] [13].

For the experiments where sound events had to be detected within a larger audio stream, prior to being classified, multiple types of classification errors can occur. It’s possible for a sound event to be falsely detected at a point in the audio stream where none occurs. It’s also possible for a sound event in the audio stream to not be detected. Lastly, it’s possible for a correctly detected sound event to still be wrongly classified. To account for this, a more complicated metric of performance would be needed, than for the experiments in which isolated audio events are classified [15]. The solutions which are aimed at detection and classification of an audio event within a larger audio stream of background noise which are reviewed in this paper are evaluated on the MIVIA road event data set or on the ITC-IRST and UPC-TALP data sets. The papers where the MIVIA data set is used, report the recognition rate (RR), the miss detection rate (MDR - existing SE is not detected), the error rate (ER - SE detected, but wrongly classified) and the false positive rate (FPR - SE detected where none exists). The papers where the ITC-IRST and UPC-TALP data bases are used do not report these metrics, but instead their calculated AEER (Acoustic event error rate) and AED-ACC (audio event detection accuracy) scores, which are two metrics conforming the 2006 and 2007 CLEAR conferences respectively, which take both miss detection, miss classification and false positives in account [14]. The AEER is described by equation 1, where $N$ is the number of ground truth sound events to detect, $N_{md}$ is the number of miss detected sound events, $N_e$ is the number of wrongly classified sound events and $N_{fp}$ is the number of falsely detected sound events.

$$AEER = \frac{N_{md} + N_e + N_{fp}}{N} \tag{1}$$

A description of the AED-ACC can be found in equation 2, where precision is the ratio between the number of correctly classified sound events and the number of predicted sound events and recall is the ratio between the number of correctly classified sound events and the true number of sound events.

$$ED - ACC = 2 * \frac{precision * recall}{precision + recall} \tag{2}$$

In this paper, we will report the AEER and AED-ACC for both sets of papers, because these metrics can be calculated for the MIVIA papers, based on the results provided and the size of the event data and background data in seconds. This calculation is made under the assumption that all sound events in are approximately of equal length. The MIVIA data set consists of 57 sound files of approx. 1 minute in length, containing a sequence of sound events each, with at least 4 seconds in between sound events [5]. We therefore assume no single sound event within the MIVIA data set to be longer than 1 minute. Note that no direct quantitative comparison will be made between the results obtained with the MIVIA and the ITC-IRST/UPC-TALP data sets, because that would not be valid due to the different nature of these data sets. However, for simplicity’s sake, a uniform performance metric is still preferred.

### 3.2 Results

The methods which have been tested on the RWCP data set include deep neural networks proposed by McLouglin et al. [10], a KNN using the SPD-IF proposed by Dennis et al. [3], and CNN’s proposed by Zhang et al. [16] and Phan et al. (2016) [13] as well as some HMM and SVM baseline methods which were included in these papers.
The performance of the different AEC methods on the RWCP data set is shown in table 3.2. Multi-condition experiments (denoted in table 3.2 with the abbreviation ‘MC’ under additional details) are experiments where the classifier was trained on noisy data at 10 db of reduction as well as clean data. In all other experiments, only clean data was used for training. Noisy data at all db levels of reduction was used for testing in all experiments. On average, the two layer convolutional neural network architectures from [16] (CNN) score the best (92,5%), being slightly better than one layer max-pooling CNN’s from [13] (1maxCNN - 92,25%).
The systems tested on the MIVIA data set include three different baseline bag of features methods (BoW) using different sound features, as well as the trainable COPE filters method proposed by Striscuglio et al. [15]. The systems tested on the ITC-IRST and UPC-TALP data sets include a weighted and an unweighted version of the random regresssion forest method proposed by Phan et al. (2015) [14], as well as an SVM and two HMM baseline systems.
The performance of the AED systems tested on the MIVIA road events data set and the performance of the AED systems tested on the ITC-IRST and UPC-TALP data sets can both be seen in table 3.2. In regards to the MIVIA data set, the COPE filters method outperforms all BoW methods with a recognition rate of 94 percent and an acoustic event error rate of approximately 18.7 percent, although the adaptive bag of features method has the lowest false positive rate. The weighted random regression forest method performed best for both the ITC-IRST and UPC-TALP data sets, with and AEER of 17.1 and 22.9 respectively.

| learning architecture | sound feature | add. details | clean | 20db | 10db | 0db | mean |
|---|---|---|---|---|---|---|---|
| HMM | MFCC | | 99.55 | 78.93 | 44.30 | 13.56 | 59.09 |
| | | Advanced Front End | 99.11 | 89.38 | 71.74 | 35.40 | 73.91 |
| | | MC | 97.53 | 95.43 | 91.94 | 67.17 | 88.02 |
| | Miss. ft. MFSC | imputation | 94.27 | 90.35 | 80.43 | 60.49 | 81.38 |
| | | marginalization | 93.65 | 85.55 | 74.67 | 50.08 | 75.99 |
| | Gabor | | 99.8 | 41.9 | 10.8 | 3.5 | 39.0 |
| | | MC | 99.39 | 91.33 | 92.51 | 56.48 | 84.93 |
| SVM | MFCC | | 98.5 | 28.1 | 7.0 | 2.7 | 34.1 |
| | SIF | | 91.13 | 91.10 | 90.71 | 80.95 | 88.55 |
| KNN | SPD-IF | Hellinger distance | 98.81 | 98.00 | 96.63 | 90.35 | 95.95 |
| DNN | SIF | | 98.07 | 85.07 | 67.53 | 31.20 | 70.47 |
| | | e-scaled denoised | 96.00 | 94.37 | 93.53 | 85.13 | 92.26 |
| | | e-scaled, denoised, MC | 94.7 | 95.8 | 92.2 | 87.7 | 92.6 |
| | | smoothed | 86.67 | 86.40 | 85.33 | 73.53 | 82.98 |
| CNN | SIF | | 97.33 | 97.40 | 95.67 | 83.07 | 93.37 |
| | | smoothed | 97.33 | 97.27 | 96.20 | 85.47 | 94.07 |
| | MelFb | | 97.67 | 97.53 | 94.67 | 70.27 | 90.04 |
| 1maxCNN | SIF | | 98.0 | 98.1 | 97.3 | 75.5 | 92.2 |
| | | energy augmented | 99.1 | 88.5 | 74.9 | 50.3 | 78.2 |
| | | MC | 98.4 | 98.3 | 98.2 | 97.7 | 98.2 |
| | | energy augmented, MC | 99.1 | 99.0 | 98.9 | 97.5 | 98.6 |

Table 1. Classification accuracy (%) on the RWCP data set of systems using various architectures and sound features to classify isolated sound events. The results are compiled from [3], [10], [16] and [13]. 'MC' is short for 'multi-condition experiment'.

| database | system | RR | MDR | ER | FPR | AEER | AED-ACC |
|---|---|---|---|---|---|---|---|
| MIVIA | COPE filters | 94 | 4.75 | 1.25 | 3.95 | 18.7* | 91.0* |
| | BoW - MFCC | 80.25 | 19 | 0.75 | 5.48 | 37.4* | 81.1* |
| | BoW - Bark | 75 | 21 | 4 | 10.96 | 60.2* | 71.3* |
| | BoW - adaptive | 82 | 17.75 | 0.25 | 2.85 | 27.4* | 85.8* |
| ITC-IRST | Regr. forest weighted | n.a. | | | | 17.1 | 91.8 |
| | Regr. forest unw. | n.a. | | | | 18.5 | 90.7 |
| | UPC SVM | n.a. | | | | 64.6 | n.a. |
| | CDU HMM | n.a. | | | | 45.2 | n.a. |
| | ITC HMM | n.a. | | | | 23.6 | n.a. |
| UPC-TALP | Regr. forest weighted | n.a. | | | | 22.9 | 89.1 |
| | Regr. forest unw. | n.a. | | | | 24.7 | 90.4 |
| | UPC SVM | n.a. | | | | 58.9 | n.a. |
| | CDU HMM | n.a. | | | | 52.5 | n.a. |
| | ITC HMM | n.a. | | | | 33.7 | n.a. |

Table 2. Detection and classification performance of sound events within a larger audio stream for the systems tested on the MIVIA data set, and ITC-IRST and UPC-TALP data sets. Results are taken from [15], [5] and [14]. *AEER and AED-ACC scores for the MIVIA systems are calculated based on the other metrics provided, as explained in section 3.1.2

## 4 Discussion

Out of all the methods tested on the data set taken from the RWCP, the multi-condition 1-convolutional layer max-pooling networks proposed by Phan et al. [13] were the most successful architectures in terms of mean classification accuracy, obtaining 98.6% with energy augmenting, and 98.2% without. Interestingly, a relatively simple K nearest neighbor classifier was the second most successful architecture when trained on an image feature obtained from the sub-band power distribution (SPD-IF), with a mean class. acc. of 96.0%.

Another noteworthy thing about the results from the RWCP data set is that multi-condition classifiers outperformed their clean condition counterparts not only for noisy conditions but on average as well. This holds for the HMM based classifiers trained on MFCC's (59.1% versus 81.4%), the HMM based classifiers trained on Gabor features (39.0% versus 84.9%) for the DNN classifiers, although only barely (92.3% versus 92.6%) and lastly also for the 1max-CNN classifiers (92.2% versus 98.2% without energy augmenting and 78.2% versus 98.6% with energy augmenting). Based on these results it seems to be the case that it is best to train a classifier on noisy data, when it will be used in a noisy environment. This conclusion (regarding CNN's) is also supported by recent research in image processing [12]. A possible explanation for the difference between the relatively small improvement of 0.3% found for the multi-condition experiments by McLouglin et al. and the larger improvements reported overall both by Dennis et al. [3] and by Phan et al. [13] could be that McLoughlin et al. only used one background noise condition, 'Speech babble', in the training of their multi-condition experiments [10], whereas Dennis et al. and Phan et al. included three background noise conditions in addition to clean data. Further research on multi-condition DNN's trained on four background conditions would be necessary in order to confirm this.

In order to investigate how the biologically inspired COPE filters by Strisciuglio et al. [15] perform compared to the random regression forests by Phan et al. (2015) [14], these would need to be evaluated on the same data set. Such a comparison would be interesting, because the outcome of it could form an argument for or against biologically inspired engineering within the AED field. On the one hand, the human hearing system is limited to certain frequencies, and it is not our strongest sense. Adhering too strictly to a biologically inspired model, may actually cause some of the limitations of our biological ears and brains to be built into the resulting classifier. On the other hand it would make sense to not treat all differences in frequencies equally and focus on the frequencies which are distinguishable by humans, when classifying sound events which are easily distinguished by humans and relevant to everyday real life AED applications.

From a biological engineering perspective it's also interesting that Aytar et al. [2] were able to train an audio CNN on unlabeled video data, using only a trained image CNN as teacher-student relationship. In the human brain, auditory and visual perception is strongly integrated, especially for non-speech sounds. Sometimes our attention can be caught by a sound within an environment, which we detect the direction of. This allows us to visually identify the object from which the sound originated, which in turn allows us to identify the sound itself.

### 4.1 Future work

Results from Dennis et al. [3] indicate that the SPD-IF significantly outperformed compared to a SIF when using a simple K nearest neighbor classifier. Furthermore, the K nearest neighbor classifier with a SIF was significantly outperformed by convolutional neural network classifiers trained on the spectrogram image, with the 1-conv. layer max-pooling networks of Phan et al.[13] being the most successful. It would be interesting to investigate whether an even better performance could be achieved when training a 1-conv. layer CNN on the sub-band power distribution image, instead of the spectrogram image. We can not be certain at this point, whether this will indeed be the case. Combining two different elements of an AEC system (the input feature and the classifier architecture in this case) which individually both improve the result compared to a baseline would logically lead one to expect the result to improve even more. The underlying assumption being that an SPD of a sound event would be a more informative representation than its spectrogram, and independently of this, a 1-layer max-pooling CNN would be better suited for distinguishing between different sound events than a simple K nearest neighbor classifier. However, according to Dennis et al. the reason why they expected the SPD-IF to outperform the SIF is because the SPD transform localizes the reliable signal components to a certain region of the image, and CNN's have the known property of being relatively shift invariant.

Regardless of this, attempting to maximize the validation accuracy on this benchmark data set of 50 different sound classes is a worthy topic of investigation, because the accuracy of any classifier typically needs to be very high, for the classifier to be viable for many commercial applications. 95% val. accuracy sounds like a lot, but that means the classifier will make a mistake once every 20 tries on average, which might already be too much in some instances.

Another area of future work lies in the application of the most successful sound event classification methods, in a sound event detection pipeline. The regression forest AED method proposed by Phan et al. [14] relies on a SVM multi-class classifier. It would be interesting to see if the results Phan et al. obtained on their AED pipeline could be improved with the use of a more complex architecture such as a 1-conv. max-pooling layer CNN or even with the use of a different feature such as the SPD-IF in the classification step.

## 5 Conclusion

In this paper, several recent developments in the field of automated audio event classification and detection have been reviewed. The focus has been on the sound features and machine learning architectures that are used to train sound event detectors and classifiers. With regard to architectures it was found that CNN's are a promising technique in this field, especially 1-conv. layer max-pooling CNN's. However, we also found a relatively simple KNN classifier could perform very well when sophisticated feature extraction was used, namely the extraction of the SPD-IF. Across multiple studies we found training on noisy data to be beneficial for classification of noisy test conditions, such as might occur in dynamic real life AED scenarios. Finally, we came across some open-ended questions regarding the effectiveness of biologically inspired engineering for AED, and the best way forward when moving from the classification of sound events to their precise detection within a larger audio stream and localization in time. One of the obstacles encountered when evaluating this was the lack of standardization in performance metrics and benchmark data sets.

## References

[1] M. J. Alam, P. Kenny, and D. D. O'Shaughnessy. Speech recognition using regularized minimum variance distortionless response spectrum estimation-based cepstral features. *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 8071–8075, 2013.

[2] Y. Aytar, C. Vondrick, and A. Torralba. Soundnet: Learning sound representations from unlabeled video. In *Advances in Neural Information Processing Systems*, 2016.

[3] J. Dennis, H. D. Tran, and E. S. Chng. Image feature representation of the subband power distribution for robust sound event classification. *IEEE Transactions on Audio, Speech, and Language Processing*, 21(2):367–377, 2013.

[4] E. C. F. P. Emanuele Principi, Stefano Squartini. Acoustic template-matching for automatic emergency state detection: An elm based algorithm. *Neurocomputing*, 149:426–434, 2015.

[5] P. Foggia, N. Petkov, A. Saggese, N. Strisciuglio, and M. Vento. Audio surveillance of roads: A system for detecting anomalous sounds. *IEEE Transactions on Intelligent Transportation Systems*, 17(1):279–288, 2016.

[6] R. Grzeszick, A. Plinge, G. A. Fink, R. Grzeszick, A. Plinge, and G. A. Fink. Bag-of-features methods for acoustic event detection and classification. *IEEE/ACM Transactions on Audio, Speech and Language Processing (TASLP)*, 25(6):1242–1252, 2017.

[7] A. G.-A. Jimmy Ludea-Choez. Acoustic event classification using spectral band selection and non-negative matrix factorization-based features. In *Expert Systems with Applications Volume 46*, pages 77–86, 2016.

[8] Y. LeCun, Y. Bengio, et al. Convolutional networks for images, speech, and time series. *The handbook of brain theory and neural networks*, 3361(10):1995, 1995.

[9] J. Ludena-Choez and A. Gallardo-Antolín. Acoustic event classification using spectral band selection and non-negative matrix factorization-based features. *Expert Systems with Applications*, 46:77–86, 2016.

[10] I. McLoughlin, H. Zhang, Z. Xie, Y. Song, and W. Xiao. Robust sound event classification using deep neural networks. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 23(3):540–552, 2015.

[11] S. Mun, S. Shon, W. Kim, D. Han, and H. Ko. Deep neural network based learning and transferring mid-level audio features for acoustic scene classification. In *2017 IEEE International Conference on Acoustics, Speech, and Signal Processing, ICASSP 2017 Proceedings*, pages 796–800. Institute of Electrical and Electronics Engineers Inc., 6 2017.

[12] T. S. Nazaré, G. B. P. da Costa, W. A. Contato, and M. Ponti. Deep convolutional neural networks and noisy images. In *Iberoamerican Congress on Pattern Recognition*, pages 416–424. Springer, 2017.

[13] H. Phan, L. Hertel, M. Maass, and A. Mertins. Robust audio event recognition with 1-max pooling convolutional neural networks. *arXiv preprint arXiv:1604.06338*, 2016.

[14] H. Phan, M. Maaß, R. Mazur, and A. Mertins. Random regression forests for acoustic event detection and classification. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 23(1):20–31, 2015.

[15] N. Strisciuglio, M. Vento, and N. Petkov. Bio-inspired filters for audio analysis. In *International Workshop on Brain-Inspired Computing*, pages 101–115. Springer, 2015.

[16] H. Zhang, I. McLoughlin, and Y. Song. Robust sound event recognition using convolutional neural networks. In *Acoustics, Speech and Signal Processing (ICASSP), 2015 IEEE International Conference on*, pages 559–563. IEEE, 2015.

# Face Recognition from Low Resolution Images: A Comparative Study

Marc Babtist and Sebastian Wehkamp

**Abstract**—Surveillance cameras generally have a poor resolution. The face images captured by these cameras tend to be of low quality, in various poses and with differing illumination. Recognizing faces under these conditions is a difficult problem. In recent years, several new solutions for face recognition in low resolution images have been introduced. These solutions compensate for the quality, poses and illumination using different methods. This paper performs a comparative study of four of these state of the art solutions, along with the baseline super resolution method. The four methods to compare differ strongly in ways of approaching the problem. The techniques covered in this paper consist of a method that uses Multi-HOG, one that utilizes sparse coding of local features, another which uses multidimensional scaling based on tensors and finally an enhancement of super resolution using relationship learning. Over the course of the paper, we will explain how these different methods work, what the drawbacks and benefits to each of them are and how they compare to one another in terms of accuracy. The final conclusion of our paper indicates that RSLR has the best performance, followed by MDS and SCLF.

**Index Terms**—Face recognition, very low resolution (VLR), low-resolution matching, super resolution, multi-HOG, sparse coding of local features, tensor-based multidimensional scaling, relationship-learning-based super resolution.

## 1 Introduction

In face recognition, one seeks to match a set of gallery images to a probe image. The performance of automated face recognition systems deteriorates significantly when the resolution of input images degrades. Surveillance camera footage tends to be of low quality; often face images are made with a low resolution (LR) like 12x12, with varying illumination conditions and poses. These LR probe images then need to be matched with high resolution (HR) gallery images. A real world example case where this occurs would be criminal investigations, in which we want to compare HR mug-shots to LR face images captured by surveillance cameras. Some of these methods make the image enhancements as seen in fictional crime dramas an actual reality.

In recent years there have been some significant developments in solving this problem. In this paper we will review some state of the art methods and compare the performance. In order to provide the reader with a good overview of the different techniques, we will discuss the benefits and limitations for each algorithm.

The commonly used approach for matching a LR image with an HR image is to use **super resolution** (SR). SR creates a higher resolution image from the probe image with the primary goal of creating a good visual reconstruction. The problem is that the various SR methods are not designed with a recognition perspective in mind. Since SR is still a common approach, we consider it as the baseline method to which we will compare other methods.

The first method we cover is the **Multi-HOG** method. This algorithm is based on a combination of different histograms of oriented gradients (HOG). Two distance measures are proposed to calculate the distance between the LR probe image and the HR gallery image. To cope with the various poses a new technique is provided for finding the most similar region. A combination of the Multi-HOG, *distance measure*, and the *most similar region* techniques promises to deliver state of the art results. [8]

The second paper extracts local features from the LR probe image and projects them in a feature space. Linear regression is used to compare the projected features with the HR image. In this paper, we will refer to this method as **Sparse Coding of Local Features** or SCLF for short. [12]

The third algorithm utilizes multidimensional scaling to transform both the HR and LR images. Tensor analysis is used to localize facial landmarks in order to compute the features to scale to. We will call this manner of LR face recognition **Tensor-based Multidimensional Scaling** or TMS. [3]

The last paper in the comparison improves on the SR method by introducing relationships. The paper proposes an approach which learns the relationship between the HR image space and the LR image space using SR. This algorithm will be referred to as **Relationship-learning-based Super Resolution** or RLSR. [13]

After a more in-depth explanation of these methods in Chapter 2, we explain the datasets used by the various papers in Chapter 3. In the same chapter, we compare the results the original authors obtained on similar datasets. To compare the papers we will lay out the performance data of the methods described, while keeping the characteristics of the datasets they were run on in mind. This should provide a good view on what their strengths and weaknesses are. In Chapter 4, we show the results of these comparisons, while we draw conclusions about them in Chapter 5. Finally, in Chapter 6, we discuss on how to improve the algorithms and further research that can be done.

## 2 Methods

In this section we will discuss all of the methods together with their respective benefits and limitations.

### 2.1 Super Resolution

Here, we discuss the Super Resolution (SR) method. SR is a commonly used approach for solving the very low resolution face recognition problem and is considered a baseline method.

#### 2.1.1 Basic method

The basic idea behind SR is to combine non-redundant information of multiple LR images into a single HR image. Closely related to SR is the image interpolation approach in which the resolution of an image is increased using a single image. However, due to the lack of extra information, the quality of the HR image is relatively low. Therefore, when performing SR we use multiple LR images. Super resolution methods can be categorized into two separate approaches. The first approach is based on statistics. Both the movement and the blurring can be regarded as stochastic variables. Therefore the SR

- *Marc Babtist is a Master student of Computing Science at the Rijksuniversiteit Groningen, E-mail: marcbabtist@gmail.com.*
- *Sebastian Wehkamp is a Master student of Computing Science at the Rijksuniversiteit Groningen, E-mail: sebastianwehkamp@gmail.com.*

reconstruction can be cast to a full Bayesian framework in which you try to maximize conditional probability. [15][13]

The second approach is example-based in which you try to make a linear combination of the LR images. First the weights are determined using a set of HR training images by trying to minimize the error. Using these weights, LR images are combined into a single HR image. [13]

### 2.1.2 Benefits

Although SR is commonly used in LR face recognition, the method has very few benefits without modifications. It is one of the earliest methods which tries to solve the LR face recognition problem, which is why we consider it the baseline method in this comparative study.

### 2.1.3 Weaknesses and limitations

The first problem of the SR method is image registration. Image registration is the process of transforming separate images into the same coordinate system. This is a fundamental image processing problem which is very hard to solve. The problem is made even more difficult because of the images having a low resolution. Traditional SR techniques treat image registration as a distinct process from the reconstruction. Therefore the quality of the reconstructed image depends on the quality of the previous step. [15]

Another problem is that the reconstructed face might not look like the original face or have serious artifacts (errors due to compression). Both the statistical and example based approach employ two constraints, the first of which is the data constraint $\varphi_D$. $\varphi_D$ ensures that the reconstructed image is similar to the LR image. The statistical approaches use it to model the probability and the example based approaches use it implicitly to determine the weights. However, because of the limited information being carried by the LR images this constraint does not suffice. This is why the second, algorithm-specific constraint $\varphi_S$ exists. $\varphi_S$ ensures that the reconstructed image is a face image. However $\varphi_S$ is designed for generic faces rather than for a specific individual. This might result in artifacts or unrecognizable faces. [13]

## 2.2 Multi-HOG

The method detailed in this section is Multi-Histogram of Oriented Gradients (Multi-HOG). [8]

### 2.2.1 Basic method

In order to explain this method, we first need to explain the histogram of oriented gradients (HOG). HOG is a feature extraction technique that uses gradient detectors in order to calculate oriented gradients, which are then saved in a histogram. The mathematical description of the HOG method is as follows: $G_x$ and $G_y$ represent the horizontal and vertical components of the gradients. These are computed using the intensity of the pixel $I$ at $(x,y)$, as per the following equations:

$$G_x = I(x+1,y) - I(x-1,y) \tag{1}$$

$$G_y = I(x,y+1) - I(x,y-1) \tag{2}$$

These components are then used to calculate the magnitude $M$ and angle $\theta$ of the gradient at location $x,y$:

$$M(x,y) = \sqrt{G_x^2 + G_y^2} \tag{3}$$

$$\theta_{x,y} = \tan^{-1}\frac{G_y}{G_x} \tag{4}$$

Now that we have defined the oriented gradients, we need to create a histogram. To do so, we define the value bins of the histogram as follows:

$$V(b_\theta) = \sum_{y=1}^{y_{max}}\sum_{x=1}^{x_{max}} M_{b_\theta}(x,y) \tag{5}$$

In this equation, $M_{b_\theta}$ is defined as follows:

$$M_{b_\theta}(x,y) = \begin{cases} M(x,y) & \text{if } b_\theta = \lceil \frac{\theta_{x,y}B}{2\pi} \rceil \\ 0 & \text{otherwise} \end{cases} \tag{6}$$

where $b_\theta$ is the bin for the angle $\theta$ and $B$ is the size of the bins. This translates to $V(b_\theta)$ containing the sum of all magnitudes that fit in bin $b_\theta$. With the normal HOG method, the image is divided into sub-images, for each of which a HOG will be constructed. After construction, these are concatenated and normalized, resulting in the feature vector for the image.

Multi-HOG expands upon this by creating multiple sets of sub-images, where each set has different grid dimensions and bin sizes. Then, each of these sets of HOGs are compared to the corresponding HOG for the other image by calculating the Euclidean distance between the two. These distances are then concatenated into a distance vector. A graphical explanation can be seen in figure 1.

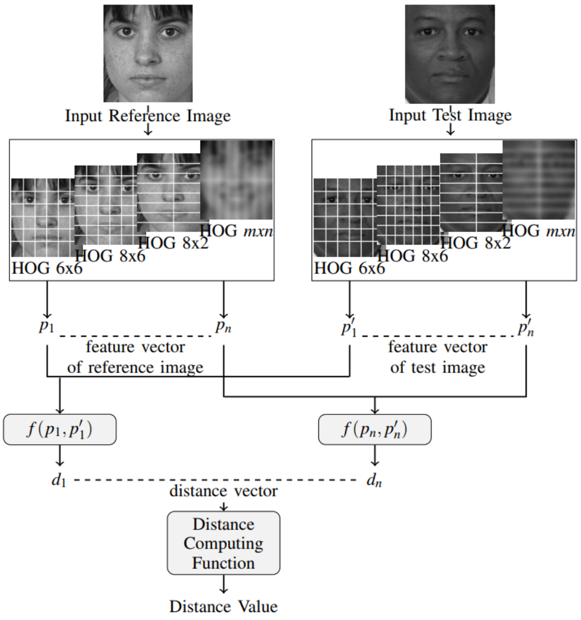


Fig. 1: Multi-HOG method of computing the distance between two faces. $f()$ is the Euclidean distance function. [8]

The resulting distance vector is then given to a distance computation function, which calculates a single distance value between the two images. After this, a 1-nearest neighbor classifier is used to conclude which face is the most similar based on the distance value. The original article experimented with two different distance computation functions: mean minimum distance (MMD) and multi-layer perceptron distance (MLPD).

During the paper's experiments, the authors also performed several general operation that increased performance. These included adding mirrored versions of the reference images to the dataset, correcting illumination differences by adjusting brightness and contrast to a fixed mean and standard deviation, and selecting the most similar regions of faces for one person.

### 2.2.2 Benefits

Since the HOG descriptor works on local cells, geometric and photometric transformations do not affect it, except for object orientation. This makes it fairly robust against changes in image quality. [4]

### 2.2.3 Weaknesses and limitations

One of the drawbacks of this method is that it is as of yet untested on image resolutions lower than 72 × 80. This means that performance on very low resolutions is unknown for this method. Besides that, calculating multiple HOGs per image makes this solution computationally intensive. [9] This is mitigated somewhat by calculating the HOGs from the reference images in advance and only once.

## 2.3 Sparse Coding of Local Features

In this section the approach of using SCLF for solving the LR face recognition problem will be discussed. [12]

### 2.3.1 Basic method

The SCLF method consists of four steps: preprocessing, extraction of local features, sparse coding, and classification. The first step is to apply some preprocessing operations to the face images. The top-hat filter is used to enhance bright objects of interest in a dark background while removing poor-contrast features. It is an excellent tool for correcting the effect of non-uniform illumination. Bottom-hat filtering is the opposite of top-hat filtering and enhances dark features of interest. The authors propose to add the difference between the top-hat and bottom-hat filtered outputs to the image. Mathematically this can be written as follows:

$$I_{CE} = I + I_{TH} - I_{BH} \tag{7}$$

where $I_{TH}$ and $I_{BH}$ are the top and bottom-hat filtered images respectively. The resulting image $I_{CE}$ is a contrast enhanced version of the original image $I$. An example of the contrast enhanced image can be seen in Figure 2.

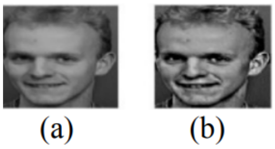


Fig. 2: (a) Original image (b) Contrast enhanced version of the original image [12]

The second step is to extract local features from the face images. Due to the face images being captured in an unconstrained environment, mistakes caused by pose variations, varying illumination, and different face expressions can occur. A robust solution to this problem is to look at descriptors of image patches: local features. The image is split in sub-images using spatial-frequency analysis which provides useful information about the image's structure at various scales and orientations. The key idea is to use Gabor wavelets as a multiscale operator using both the spatial and the frequency domain.

The third step is to create a sparse representation from each face image. Sparse representation has proven to be an efficient technique for face recognition. For every face a sparse representation is made with a linear combination of other training samples. The paper proposes a new way to create such a sparse representation by finding an optimal projection matrix, which maps local features to a sparse low-dimensional feature space.

After projecting, the last step is to perform classification using linear regression. This is done by calculating the residual errors of representing a testing feature in terms of training features. The image is assigned to the class with the lowest reconstruction error. A flow chart of the proposed approach can be found in Figure 3. [12]

### 2.3.2 Benefits

In general, the advantage of classification using sparse representation is that it gives very good results when images are heavily corrupted by noise or occlusions. This allows for good results in practical face recognition problems. [7] Another advantage of the proposed method is that it is more concise. Only the sparse representations of the images are needed for comparison instead of the complete image, meaning this method requires relativley little storage space. [14]

### 2.3.3 Weaknesses and limitations

The training images must be carefully controlled and sufficient training samples of each class must be present or else the performance is affected badly. [7]

## 2.4 Tensor-based Multidimensional Scaling

In this section the approach of using MDS for solving the LR face recognition problem will be discussed. [3]

### 2.4.1 Basic method

The article [3] proposes a multidimensional scaling based approach which transforms features from both the LR and HR images to a common space such that distances can be calculated. The features are retrieved from the images using the Scale Invariant Feature Transform (SIFT) which is robust to changes of scale and rotations. These features should be transformed to a common space using a transformation. In this common space the distance $D_t$ between the transformed HR image and the transformed LR image should equal the distance between the images had they been the same resolution and pose. To determine this transformation, training data consisting of frontal HR images is used. Tensor (vector) analysis is performed on the LR probe image to estimate the locations of facial landmarks. A flow chart of the proposed approach can be found in Figure 4. [3]

### 2.4.2 Benefits

Due to the automatic feature localization using tensor analysis, the algorithm performs very well in tracking and recognition in surveillance videos. This can be done simultaneously with a couple of modifications. Because of the tensor analysis which localizes features, the algorithm should also be able to handle changes in pose well. [3] [7]

### 2.4.3 Weaknesses and limitations

SIFT is based on the HOG of each pixel which makes it computationally intensive. Although the paper proposes to use PCA in combination with SIFT to alleviate this problem. [9]

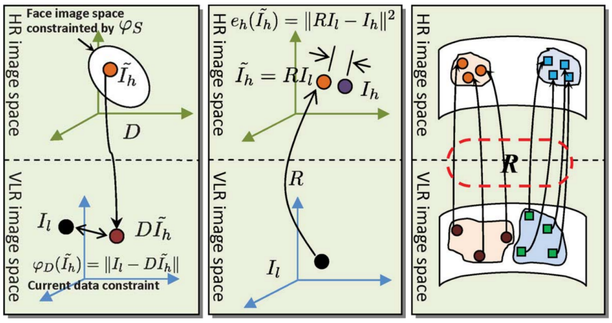


Fig. 5: Comparison of error calculation methods for human-based recognition. Left is the error calculation of existing approaches. The middle is the error calculation of the proposed approach. The right is the machine-based recognition approach. [13]

## 2.5 Relationship-learning-based Super Resolution

In this section the approach of using RLSR for solving the LR face recognition problem will be discussed. [13]

### 2.5.1 Basic method

The RLSR approach tries to find the relationship between the HR and LR image by learning it in the training phase. This relationship is used to reconstruct the HR images from the LR images in the testing phase. The first step in the training phase is to create linear clusters from LR and HR images. This means that the relationship in each cluster can be represented by a matrix. In the next step a linear regression model is used to learn this relationship with two different constraints. The first constraint aims at human-based recognition. This method first maps the LR image to an HR image and then calculates the error. This

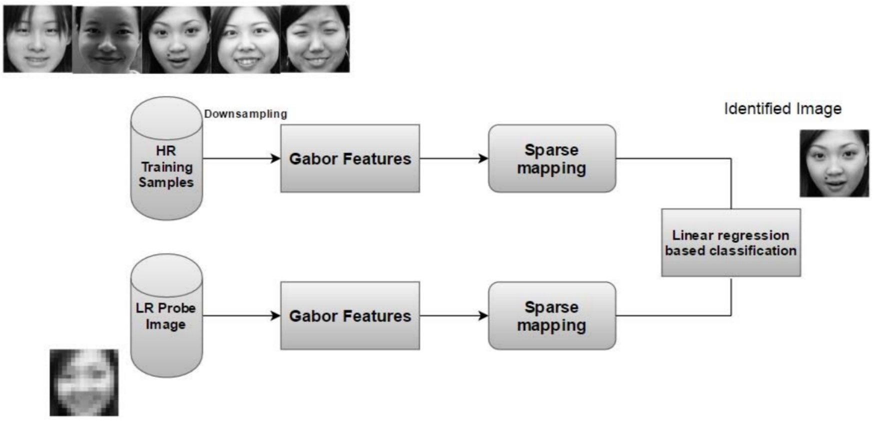


Fig. 3: Flow chart of the proposed approach. [12]

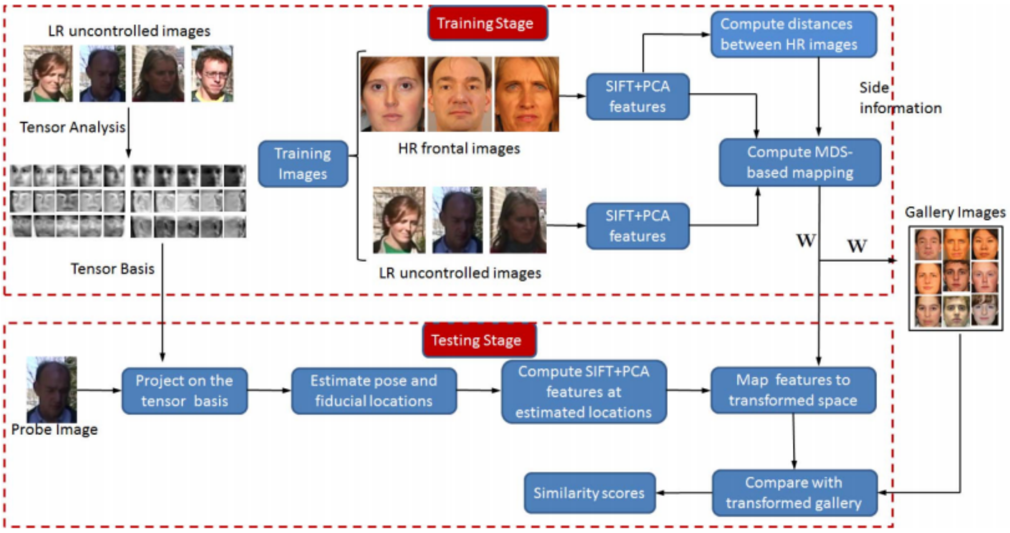

LR uncontrolled images
Tensor Analysis
Tensor Basis
Training Stage
Training Images
HR frontal images
LR uncontrolled images
SIFT+PCA features
SIFT+PCA features
Compute distances between HR images
Side information
Compute MDS-based mapping
W
W
Gallery Images
Testing Stage
Probe Image
Project on the tensor basis
Estimate pose and fiducial locations
Compute SIFT+PCA features at estimated locations
Map features to transformed space
Compare with transformed gallery
Similarity scores

Fig. 4: Flow chart of the proposed approach. [3]

is different from existing methods, which calculate the error in the LR image space. Examples of both the error calculation in existing methods and the proposed method can be seen in Figure 5. [13]

The second method is aimed at machine-based recognition. This method was designed because visual quality is not the only criterion for SR. When looking from a machine-based perspective discriminability should also be considered. The goal is to create HR images with discriminative features in contrast to creating an HR image with a small error. This is done by using their Discriminative SR (DSR) algorithm which optimizes the discriminability of different images using class label information. In Figure 5 one can see that the small differences in LR space are enhanced in HR space. [13]

### 2.5.2 Benefits

One of the benefits of this approach is that the authors thought of both machine-based and human-based recognition. Since the results of human-based recognition ought to be better than the results of standard SR, this method can be used instead of SR in other use cases than LR face recognition. The machine-based approach should emphasize the discriminative features compared to SR so that it performs better than standard SR. [13]

### 2.5.3 Weaknesses and limitations

The problem of image registration in the standard SR method still holds for this method. This means that the quality of the SR algorithm depends on the quality of the image registration.

## 3 Datasets and Experimental Setup

This section describes the various datasets used and which experiments were performed.

### 3.1 Datasets

In this section, we will perform a short comparison between the various datasets used in the articles. To see which dataset was used in which article, refer to Figure 6.

| Dataset | Multi-HOG | SCLF | MDS | RSLR |
|---|---|---|---|---|
| ORL | | x | x | |
| Yale B | | x | | x |
| CAS-PEAL | | x | | x |
| CMU-PIE | | | x | x |
| FERET | x | | | |

Fig. 6: Table showing what datasets were used in each article.

### 3.1.1 ORL

The Olivetti Research Laboratory [2] (ORL) database from AT&T in collaboration with Cambridge University is a set of four hundred images, containing ten different images of each of forty subjects. Some of the images were taken at different times, varying the lighting, facial expressions and accessories such as glasses. There is no variation in background or position.

### 3.1.2 Yale B

The Extended Yale B database [6] contains images of 28 subjects under 9 angles and 64 illumination conditions, for a total of 16128 images. Facial expressions do not differ between images.

### 3.1.3 CAS-PEAL

The Chinese Academy of Sciences - Pose, Expression, Accessory and Lighting database and its subset CAS-PEAL-R1 [5] contain many images of a large amount of individuals (99,594 of 1040 and 30,900 of 1040 respectively). The set boasts a large amount of different poses, expressions, accessories and lighting situations, including combinations of the four. One thing to note is that this dataset contains mostly people of Chinese ancestry, while the other databases tend to contain more Caucasian faces.

### 3.1.4 CMU-PIE

The Carnegie Mellon University Multi-Pose, Illumination, Expression Face Database (CMU-PIE) [1] is an improvement over the earlier PIE database. It contains 337 subjects, each captured under 15 view points and 19 illumination conditions in four recording sessions, for a total of more than 750,000 images. Of these images, 6152 are provided with labels for feature points of the faces.

### 3.1.5 FERET

The Facial Recognition Technology dataset [11] [10] was developed by DARPA specifically for the evaluation of facial recognition algorithms. It contains 1564 sets of images for a total of 14,126, with varying illumination, facial expressions, angles and accessories.

## 3.2 Performance Comparisons

The goal of our paper is to select the best performing face recognition method in different situations. The problem is that the four papers selected different datasets to test their algorithms. To be able to select the best algorithm we will perform three comparisons. The first one is comparing Sparse Coding of Local features with Tensor-based Multidimensional Scaling using the Yale-B and the CAS-PEAL datasets. The second is between MDS and RLSR using the Multi-PIE dataset. The last comparison is between Multi-HOG, SCLF, and MDS, using ORL and FERET. The datasets of both FERET and ORL can be considered similar enough to directly compare on, since they both contain images with varying illumination, facial expressions, angles, and accessories.

# 4 Results

In this chapter, we will compare the performance of the different methods, keeping in mind the datasets the performance was measured on. All of the data was obtained from the corresponding papers. In all comparisons SR is included as a baseline method.

## 4.1 Comparison 1

First we compared SCLF and MDS on Yale-B and CAS-PEAL. The results can be found in Figure 7 and Figure 8. The SCLF results were obtained by using LR images of 20x18 and gallery images of 60x55. The MDS results were obtained by using 20x20 images for CAS-PEAL and 16x16 for Yale-B. We used the results obtained by using four training samples. The testing conditions are very comparable, which means that the results are, too. In both comparisons SCLF and MDS perform very similar but SCLF has a slight edge over MDS.

## 4.2 Comparison 2

This experiment compares RLSR and MDS using the CMU-PIE dataset. The RLSR results were obtained using a very low resolution of 7x6 and the gallery had a resolution of 56x48. These results were taken from rank 6. This rank indicates that the result came from the sixth worst training/test set, in this case out of ten. This was chosen because we feel it best represents the capabilities of the method.

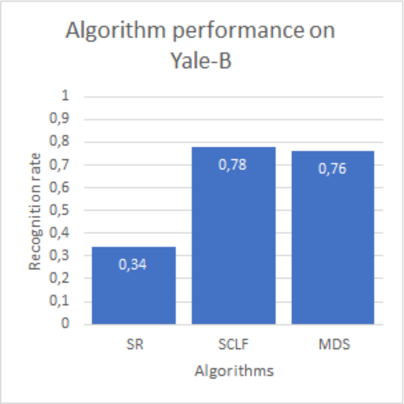


Fig. 7: Performance of various algorithms on the Yale-B dataset.

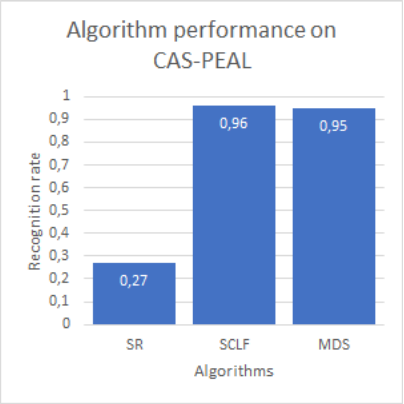


Fig. 8: Performance of various algorithms on the CAS dataset.

The MDS results are obtained using LR images of 13x11 while the gallery images have a resolution of 60x55. The results can be found in Figure 9. RLSR clearly performs significantly better than MDS even though the LR images of RLSR have a lower resolution than the LR images of MDS.

## 4.3 Comparison 3

The last comparison is done on both the ORL and FERET datasets, which are very similar to each other. The MDS and SCLF results come from the ORL dataset while the Multi-HOG results come from FERET. Multi-HOG uses gallery images of 80x88 but it is unclear what the resolution of the LR images is. We used the results of the best performing variant of Multi-HOG, which is Multi-HOG MSRS-MLPD. Both the MDS and SCLF results are obtained using gallery images of 60x55 and LR images of 16x16 using 4 training samples. In Figure 10 one can see that, as in comparison 1, MDS and SCLF perform very similarly with SCLF coming slightly ahead. They both perform better than Multi-HOG. The addition of mirrored images does give a minor increase to the performance of Multi-HOG.

# 5 Conclusion

All of the tested methods perform significantly better than the baseline SR method. It is clear that MDS and SCLF perform very similarly, with SCLF coming slightly ahead of MDS. We only have results of one dataset for Multi-HOG, but it comes in last in that case by a margin, though it should be noted that the exact testing conditions such as the resolution of the LR images remain unclear. From the Multi-HOG results we can conclude that adding mirrored faces does increase the performance. From comparison 2 it becomes clear that RLSR performs much better than MDS. Judging from the gap between RLSR

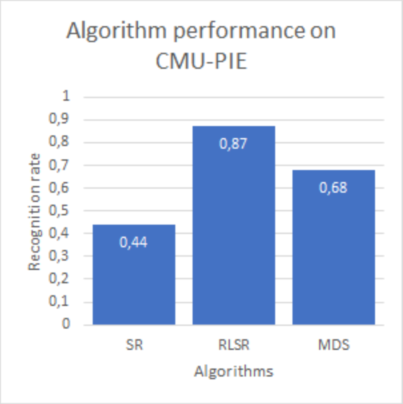


Fig. 9: Performance of various algorithms on the CMU-PIE dataset.

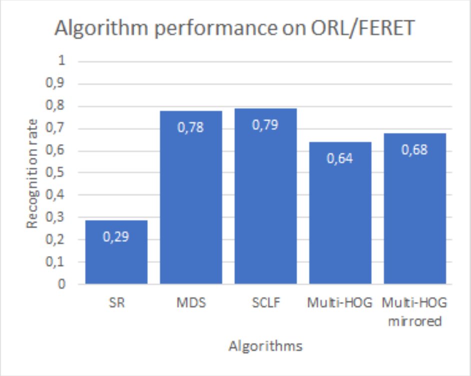


Fig. 10: Performance of various algorithms on the ORL and FERET dataset.

and MDS in comparison 3, and MDS and SCLF performing very similarly, could indicate that RLSR also performs better than SCLF.

## 6 Discussion

Here, we will suggest general improvements to the various algorithms as well as propose further research that can be performed on the subject of low resolution face recognition.

### 6.1 General Improvements

Some of the articles used in this paper contain improvements that can work on all algorithms. These include the following:

- adding mirrored faces [8],
- correcting the illumination by adjusting brightness and contrast to fit a mean and standard deviation [8],
- top- and bottom-hat filtering [12].

By performing these preprocessing operations, the set of reference images can be enhanced and expanded, providing increased accuracy.

### 6.2 Further research

The biggest flaw in the manner of comparison between the various face recognition algorithms is that we were unable to test the various methods on the same datasets due to time constraints. This means that the raw accuracy numbers obtained might not be strictly comparable. Therefore, we suggest a follow-up article that compares the various prescribed methods by running them in the same manner on the same datasets. This should provide numbers that can be compared as-is, without further alteration. However, one should take care to pick a greater variety of datasets, as certain algorithms might outperform others in specific situations.

Another subject of research could be combining the various algorithms. These could either be naively combined by voting, or by more complicated methods such as using the automatic feature localization from [3] to select the maximum similarity region for Multi-HOG. We feel that by combining the various strengths of the different methods covered in this paper a new algorithm can be created that is better than the sum of its parts.

## References

[1] The cmu multi-pie face database. http://www.cs.cmu.edu/afs/cs/project/PIE/MultiPie/Multi-Pie/Home.html. Accessed: 2018-04-11.

[2] AT&T Laboratories Cambridge. The database of faces. http://www.cl.cam.ac.uk/research/dtg/attarchive/facedatabase.html, 2002. Accessed: 2018-04-11.

[3] S. Biswas, G. Aggarwal, and P. J. Flynn. Pose-robust recognition of low-resolution face images. *Cvpr 2011*, 2011.

[4] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. *International Conference on Computer Vision & Pattern Recognition*, 1:886–893, June 2005.

[5] W. Gao, B. Cao, S. Shan, X. Chen, D. Zhou, X. Zhang, and D. Zhao. The cas-peal large-scale chinese face database and baseline evaluations. *IEEE Trans. on System Man, and Cybernetics (Part A)*, 38(1):149–161, Jan. 2008.

[6] A. Georghiades, P. Belhumeur, and D. Kriegman. From few to many: Illumination cone models for face recognition under variable lighting and pose. *IEEE Trans. Pattern Anal. Mach. Intelligence*, 23(6):643–660, 2001.

[7] S. Habeeba and P. Simon. Review on sparse based face recognition methods in uncontrolled environment. *International Journal on Computer Science and Engineering (IJCSE)*, 9(8), 2017.

[8] M. Karaaba, O. Surinta, L. Schomaker, and M. A. Wiering. Robust face recognition by computing distances from multiple histograms of oriented gradients. *2015 IEEE Symposium Series on Computational Intelligence*, 2015.

[9] Y. Ke and R. Sukthankar. Pca-sift: a more distinctive representation for local image descriptors. *Proceedings of the 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2004. CVPR 2004.*, 2004.

[10] P. Phillips, H. Moon, S. Rizvi, and P. Rauss. The feret evaluation methodology for face recognition algorithms. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 22:1090–1104, 2000.

[11] P. Phillips, H. Wechsler, J. Huang, and P. Rauss. The feret database and evaluation procedure for face recognition algorithms. *Image and Vision Computing J*, 16(5):295–306, 1998.

[12] M. S. Shakeel and Kin-Man-Lam. Recognition of low-resolution face images using sparse coding of local features. *2016 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA)*, 2016.

[13] W. W. Z. Wilman and P. C. Yuen. Very low resolution face recognition problem. *2010 Fourth IEEE International Conference on Biometrics: Theory, Applications and Systems (BTAS)*, 2010.

[14] J. Wright, A. Y. Yang, Arvind, S. S. Sastry, and Y. Ma. Face recognition via sparse representation, Mar 2008.

[15] J. Yang and T. K. Huang. Image super-resolution: Historical overview and future challenges. 2010.

# A comparison of state-of-the-art vision-based biometric analysis methods

Tinco Boekestijn, Roel Visser

**Abstract**—
In this paper we perform a literature review of three state-of-the-art vision-based biometric analysis methods. The first is a fingerprint biometric method based on a Minutia Cylinder-Code (MCC) method. The second, a palmprint biometric analysis method which uses a dynamical system approach. The last method is based on retinal vascular networks using linear comparison to tolerate for scale, rotation, and translation. Each of the biometric analysis methods aims at efficient, accurate identification. We compare the advantages and disadvantages of each method according to their usability, characteristics, and performance. The MCC method improves greatly over previous fingerprint methods in both storage size and speed of the matching algorithm. It also stacks up very well against the other two methods in terms of speed, accuracy, and collectability. This method is the most useful in applications where identification speed should be high and a large dataset has to be checked. The palmprint method has a clear advantage with respect to the low-resolution of images that it can use for identification. The usage of linear comparison with a number of tolerances of the retinal vascular analysis method give it a clear advantage over previous methods in terms of matching speed and the ability to tolerate scale, rotation, and translation. Its speed however is prohibitive in it being a competitor to fingerprint and palmprint in cases that require high performance. If security is the most important factor however this method is the best of the three.

**Index Terms**—Biometrics, fingerprint identification, retinal vascular identification, palmprint identification.

## 1 Introduction

Biometric analysis is a critical method in providing identification based on *'things you are'* and not on *'things you know'*, such as a password or personal identification number.

There exist many established biometric analysis solutions that are used in a wide range of applications such as forensic research, authentication, and encryption. Currently, such biometric analysis methods are being researched using pattern recognition and machine learning in order to achieve a high success rate on verification and identification. In many cases these solutions should be compliant with the strict requirements of (high) security applications.

There are various methods to identify a person based on their unique traits. In this paper we will focus on vision-based biometric methods. We will look at three state-of-the-art biometric analysis methods. One that uses retinal vasculature, one using palmprints, and another using fingerprints. Each of these is vision-based since these features can be extracted using image processing methods. Other biometric analysis based on features such as voice, gait, or other behavioral methods are not included in this paper.

In this paper we will try to compare these state-of-the-art biometric analysis methods in order to determine their advantages and disadvantages and to see in what way each of the methods improves over other methods within their field. Because we will be comparing methods from different fields of biometrics analysis, we start off by giving general background on each of these fields (fingerprint, palmprint, and retinal vascular) in the next section. In this way we will be able to make a comparison between methods that are based on very different biometrics features. In order to fully appreciate the contextual differences of each method, it is, for example, important to know that retinal vascular imaging requires very different cameras than are required for obtaining fingerprints.

Once the background of each field is covered, we will look into the three methods. By doing this we want to be able to both determine in what way these methods improve over previous ones and create a complete picture of each method. When we have gained a comprehensive overview of the fields and the state-of-the-art methods we will be able to make a comparison between the three in the comparison section (sec. 4).

In the comparison we focus on properties that allow us to determine the pros and cons of different methods and in which use cases they might be applicable.

We base our comparison on the following characteristics/features:

- Universality: every person should have the trait
- Uniqueness: no two persons should be expected to have an identical biometric trait
- Permanence: a trait should not vary over time.
- Collectability: the ease of collection from an individual.
- Performance: the identification accuracy, speed, and robustness under varied environmental circumstances.
- Acceptability: the acceptance of the usage of the technique on the relevant population.
- Circumvention: the ease of use of fraudulent techniques to fool the system. The biometric should be difficult to deceive.

The chosen biometric characteristics are defined in the Handbook of Biometrics for Forensic Science [9] and used in Meng et al. [15], as such we will also use these criteria in our comparison. The Background section will cover characteristics that are more generally applicable to a specific biometrics field (i.e. Universality, Uniqueness, Permanence, Collectability, Acceptability, and Circumvention). While the Methods section will be most relevant to the Performance and to a lesser extend the Circumvention characteristic of the methods. The performance characteristic is subdivided in:

- Accuracy: e.g. what is the false acceptance rate (FAR), false rejection rate (FRR), receiver operating characteristic (ROC), authentication accuracy
- Speed: how quick can images be processed into usable features and how fast is the matching/comparison process
- Robustness: how well can the system handle noise, permutations, etc. in the image.
- Storage: how efficient can extracted features be stored

## 2 Background

In this section we give a quick overview of biometric analysis and the general structure of fingerprint, palmprint, and retinal vascular network analysis. This is done in order to give insight in the pros and

- *Tinco Boekestijn is a Computing Science master student at the Rijksuniversiteit Groningen, E-mail: t.y.boekestijn@student.rug.nl*
- *Roel Visser is a Computing Science master student at the Rijksuniversiteit Groningen, E-mail: r.w.visser.1@student.rug.nl .*

cons of each of these biometric analysis methods in general. In the next section we go into the state-of-the-art methods of each of these fields.

The techniques of identification using biometrics keep evolving, however the basic structure of biometric analysis has not changed. A process diagram is displayed in figure 1 that shows the different stages of biometric identification and verification that are the main components that any biometrics system uses.

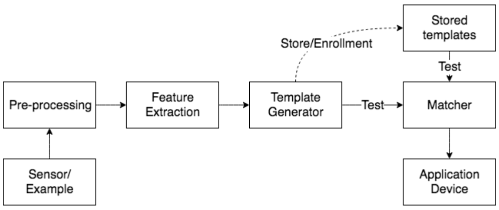


Fig. 1: The phases of biometric identification and verification. [9]

The first step of biometric analysis is retrieving biometric information from a person. This is usually done by retrieving a two-dimensional image of some part of the body (e.g. an image of someone's fingerprint). We will call this image the example or example image. The image is preprocessed to improve the quality of data. It is important that this is done properly, since the quality of the output is dependent on the quality of the input.

The preprocessed example image might contain data that is not relevant to the analysis method. To optimize storage and processing speed the important features are extracted in the follow phase. These features should keep both the loss of quality of the data and storage size at a minimum, however this is often a trade-off.

The extracted features are compared with a database of other processed examples. Each stored example is assigned a score based on the level in which its features correspond to the extracted features. The comparison is done through a matching algorithm. The matching algorithm is dependent on the data structure that is provided by the feature extraction.

There are two different usages of the biometric matching. These methods are based on the application of the biometric analysis:

- Verification whether two examples $P_1$ and $P_2$ belong to the same person. This is a 1 : 1 comparison that returns acceptance or refusal of the equality based on a similarity score from both examples.
- Identification whether the input example $P_1$ equals an example in a collection of examples $\{P_1, P_2, ..., P_N\}$. it is an $1 : N$ comparison of the examples that returns the example with the highest score $m_{best}$. The example with the highest score is rejected when the score is lower then a certain threshold $\phi$.

No single biometric method will meet every requirement of all possible applications. The availability of biometric features is also dependent on the context (e.g. forensic science can only retrieve physical biometric identifiers that are left behind, such as fingerprints or palmprints, but not things like voice samples or retinal images).

### 2.1 Fingerprint identification

Fingerprint identification is the most widely used biometric technique for identification. It is common place in many laptops and most modern smartphones. This is because finger ridge configuration (i.e. a fingerprint) does not change throughout life, except due to accidents, deliberate alteration, medication, or surgery [13]. The acquisition of fingerprint images is relatively easy. A simple camera system which can be placed even in small devices, like a smartphone, is enough to acquire images of fingerprints that are usable in biometric analysis. It is also possible to retrieve fingerprints from smooth surfaces that an individual has touched, which makes it a common tool in the field of forensics [13].

The features of fingerprints are extracted into minutiae. Minutiae are local features of fingerprint, such as ridge endings and bifurcations of ridges in the finger. These features are extracted by using a grayscale image of the fingerprint. The image is converted into a skeleton image as shown in figure 2. From this skeleton image the minutiae are extracted. The direction, location, and classification of each minutia can be used to compare fingerprints with each other.

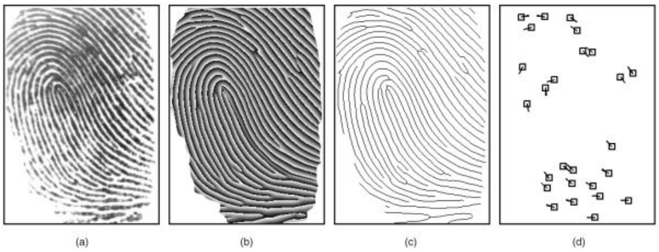


Fig. 2: Fingerprint representation schemes. (a) Grayscale image, (b) Phase image, (c) Skeleton image and (d) Minutiae [5]

Comparing two fingerprints requires finding a spatial and directional relation of a number of minutiae. The variation of features however makes this a difficult task. Modern algorithms first compare local similarities and then combine them into a global similarity [14].

### 2.2 Palmprint identification

Palmprint identification is getting considerable interest in the field of biometric identification, since it can be used with low-resolution imaging and low-cost capturing devices [16]. It is also not very intrusive to use on participants. It can be used in conjunction with fingerprint identification to get a more accurate identification. The techniques of palmprint identification are divided based on whether they use low- and/or high-resolution features.

The low-resolution approach assumes that only features such as wrinkles and the principal lines of the palm can be extracted from the image. While the high-resolution approach also includes the local minutiae. The palmprint analysis method discussed in section 3 will focus on a low-resolution features approach.

The palm of a person typically consists of three principal lines [19] as shown in figure 3a. These principal lines can still be observed below a resolution of 100 dpi [16], which makes them highly distinctive. Each principal line is a smooth curve that has an origin region and crosses certain predefined line-segments, which are shown in figure 3b.

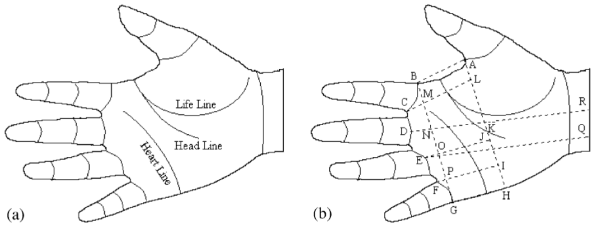


Fig. 3: (a) The typical principal lines on a palm; (b) Defined points and lines on a palm. [19]

Principal lines have some useful properties. Principal lines are still distinctive in missing and noisy data, because principal lines do not deviate from their path by a large margin. The extracted part of the lines can be used to predict the location and the direction of the next path. It also supports the thickness as an additional feature, since the thickness of the principal lines varies between persons [19].

### 2.3 Retinal vasculature identification

Besides fingerprint and palmprint, retinal vascular identification is an important biometric method. It is used for both verification and identification, especially in high security applications such as nuclear facilities or weapons factories, where extremely high security measures are needed [1, 7, 8].

Modern retinal vascular identification uses fundus cameras to obtain retinal images suitable for personal identification systems [20, 21]. These cameras are designed for imaging the back surface of the eyes, including the retina, optic disc, and macula.

Once the retinal image is obtained, feature extraction is performed. This is done by extracting the blood vessel skeleton and separating it from the background. One of the main issues in using these images for automated identification is the scaling, rotation, and translation that might occur between two images taken from the same retina.

After the features are extracted the identification process is performed. This is usually done by taking a similarity metric between the retinal image and reference images in a database.

A benefit of retinal identification is that every person has a different vascular network in their retina and even identical twins have completely different vascular networks [17].

Other than fingerprints and palmprint, retinal vascular networks remain unchanged throughout a person's lifetime and more importantly are unalterable through plastic surgery or other methods, something which is possible in finger- and palmprints [11]. Degeneration might occur however as a consequence of a disease, such as diabetic retinopathy.

A drawback of retinal identification is the need for specialized fundus cameras to obtain an image of the retinal network, which are relatively costly compared to other biometrics systems.

Overall, the use of retinal identification is a method based on very distinctive physical traits, but the complexity of most algorithms and the cost associated with obtaining the retinal images has hampered the wide spread adoption of retinal vasculature for biometrics systems.

## 3 Methods

In this section we discuss state-of-the-art biometrics methods for fingerprints, palmprints, and retinal vascular images. We focus on the most important innovative techniques that these methods employ to improve their performance compared to previous methods. We can use this information, along with performance benefits that these techniques entail, in our comparison. In the next section these methods are compared using the criteria defined in section 1.

In order to properly compare these methods and substantiate how they improve over previous methods in their respective fields, we will go slightly in-depth in some of the inner workings of image processing and matching techniques they use. In this way we will be able to gain insight in advantages and disadvantages of each method and the way the analysis of fingerprint, palmprint, and retinal vascular networks differ.

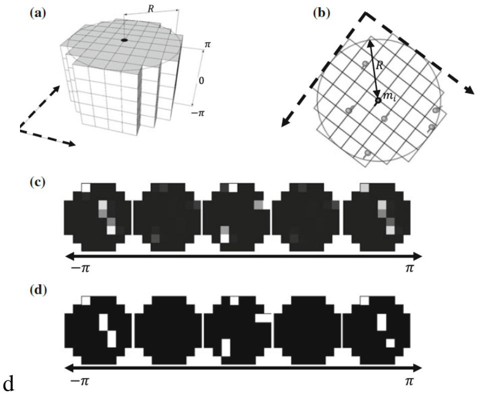
### 3.1 Fingerprint matching using the Minutia Cylinder-Code method

There are multiple state-of-the-art fingerprint identification algorithms. A Minutia Cylinder-Code (MCC) method is proposed by Raffaele Cappelli et al, that achieves better accuracy than the previous methods [4]. This method achieves higher accuracy than previous methods, since it uses 3D data structures, called cylinders, to represent the minutiae.

The main advantages of the MCC method is that it tolerates missing and spurious minutiae better than other fingerprint analysis methods [13]. It is invariant to translation and rotation by using the corresponding minutia's angle. It also supports a bit-oriented coding, which makes cylinder matching simple and fast, since the comparison method between fingerprints uses bit-wise operations.

To understand how this Cylinder-Code is build up, we first have to define some variables. The minutia $m_i$ in the MCC method is described by three parameters $(x_i; y_i; \theta_i)$. The parameters are the coordinates $(x_i, y_i)$ and minutia orientation $(\theta_i)$ in range $[0, 2\pi]$. These coordinates represent the location of the surface of the fingerprint in 3d space. The fingerprint can be represented in a minutia vector $\{m_1, m_2, \ldots, m_r\}$.

The cylinder is divided in $N_D$ layers in a range between $[-\pi, \pi]$ and the layers are discretized in $N_S$ number of cells ($N_S$ is the amount of cells in a diameter). A numerical value of each cell is calculated based on the likelihood of finding minutiae that are close to the cell and the directional difference with respect to the minutia ($m_i$) as shown in Figure 4. Finally, the likelihoods are normalized to get a binary pattern in the Cylinder-Code. In this manner the representation of a fingerprint can be stored very efficiently with a minimal loss of accuracy [3].



d

Fig. 4: The process of the MCC method (a) A graphical representation of the local descriptor of a minutia in the MCC representation; (b) A representation of the minutiae in a cylinder; (c) Cell values of the different layers of the cylinder (d); Cell values of the different layers of the cylinder binarized; The cell values of (c) and (d) are rotated according to the direction of minutia $m_1$. [13]

The similarity between two cylinders can be measured for each pixel $v$ by the formula with the $\oplus$ as the bit-wise XOR operator and the $||\cdot||$ as the Euclidean norm (a threshold of $\delta_\theta$ which is used to control the maximum rotation that is allowed between two fingerprints).

$$S_L(c_i, c_j) = \begin{cases} 1 - \frac{||v_i \oplus v_j||}{||v_i|| + ||v_j||}, & \text{if } d_\phi(\theta_i, \theta_j) \leq \delta_\theta \\ 0, & \text{otherwise.} \end{cases} \quad (1)$$

Similarity measures of each minutia have to be combined to compare two fingerprints with a global score using equation 1. The fastest implementation is the Local Similarity Sort (LSS). Which compares the global score by calculating the average of the corresponding similarities.

### 3.2 Palmprint matching using the Dynamical System approach

In the paper of David Palma [16] a method is proposed to combine the extraction of a region of interest and principal lines and wrinkles. It uses a *dynamic algorithm* that involves a positive linear dynamical system [2] to compare the result of the palmprint images.

The first step is to retrieve the region of interest (ROI) [18]. The ROI can be extracted by finding the defined points C and F of figure 3. The defined points of C and F are found by using edge detection with a Canny operator on the palmprint image. A tangent line is drawn between point C and F and the angle is used to rotate and crop to the y-axis. The square that represents the region of interest is calculated by using the distance between point C and F as height and width of the image. This process is shown in figure 5. The rest of the original image (e.g. fingers, background, wrist, etc.) is discarded.

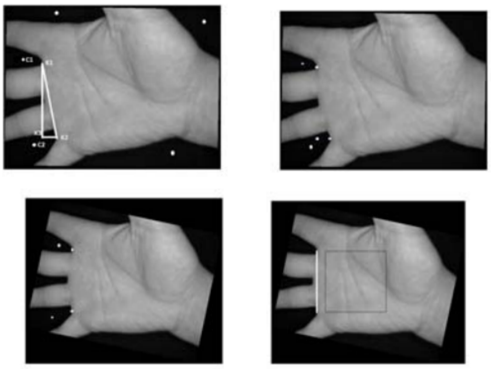


Fig. 5: The process of extracting the region of interest from a palm. [18]

Now that the ROI is extracted from the image matching between ROI's of different images can be performed. A simple approach is to compare the two images with an intersection $X \cup Y$. However a dynamic algorithm can be used to reject the noise between two different binary images. In a positive linear dynamical system a pixel is only accepted if it is exceeds a threshold of pixels $\theta$ within a radius $v$. An example of the dynamic algorithm is presented in figure 6. Individual noise and off-by-one errors are reduced, since the pixels are coupled with the surrounding neighbours.

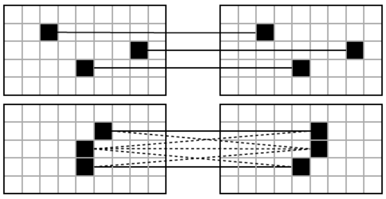


Fig. 6: Dynamic algorithm example: The points are connected through the neighbours of the surrounding point. [16]

### 3.3 Retinal vasculture matching using linear comparison

In this section we discuss the retinal identification system as proposed by Köse and İkibaş [11]. This method differs from other retinal identification systems because a method is proposed that is relatively simple, both computationally and in its implementation, yet can deal with scale, rotation, and translation of input images. Something which a lot of the earlier methods have a hard time of dealing with efficiently. This makes it an interesting method to compare with fingerprint and palmprint methods, because it might be more applicable to real time applications than other retinal vascular analysis methods.

In this method the vessel structure is extracted from an inverted 8-bit grayscale image. These images are then compared with other retinal vascular structures that are stored in a database.

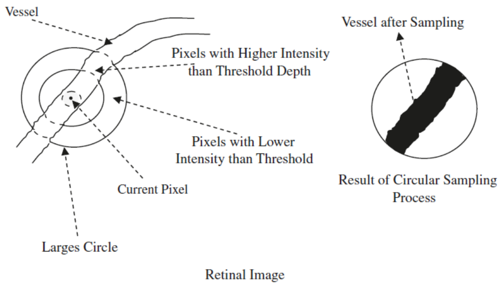

Vessel
Pixels with Higher Intensity than Threshold Depth
Vessel after Sampling
Pixels with Lower Intensity than Threshold
Current Pixel
Result of Circular Sampling Process
Larges Circle
Retinal Image

Fig. 7: Circular sampling of blood vessels. [11]

The vessel structure is segmented in the retinal images. This is done by taking a circular sampling at each pixel. The pixels in the circular area are sampled within a depth relative to the current pixel's intensity value. The result of this process is shown in Figure 8. From this the number of 'black' pixels are counted and the ratio of black pixels to the total number of pixels is calculated. If this fraction is within certain value range the pixel is classified as belonging to a vessel.

Following this, some non-vessel components might still be labeled as vessels, as shown in figure 8b. A lot of small element are visible, which can be regarded as a type of noise. To eliminate these fragments the elimination process searches for small and non-vessel fragments, such as square fragments.

This way the clean vessel structure in figure 8c is obtained.

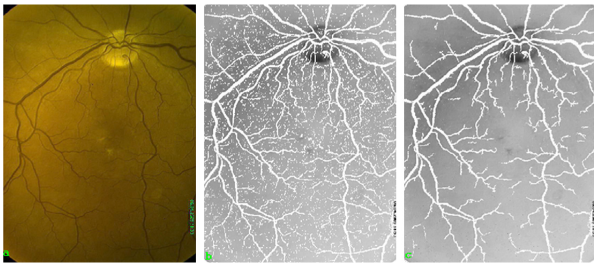


Fig. 8: Original image (a), segmentation result (b), and final segmented image after eliminating the non vessel fragments (c). [11]

Direct comparison between retinal images can be a very complex process because scaling, translation, and rotation operations are almost always required. This is due to the variability of the imaging process. A person might be in a slightly different position each time an image is taken. To this end, several simple steps are used to account for these issues.

The essential technique of the method is to count the number of vessels encountered that intersect the processing line on the image and reference images.

To make this method scale invariant, the image is sampled with lines of different scaling sizes. An example of this is shown in figure 9a.

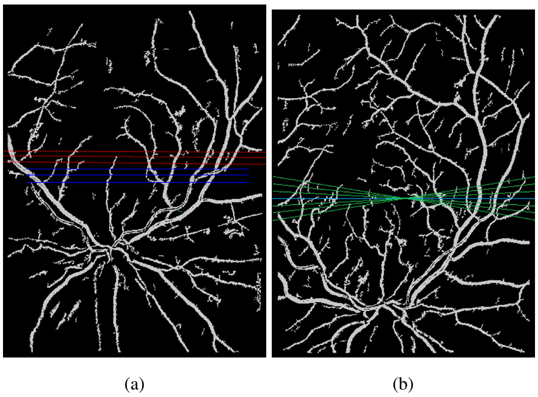


Fig. 9: (a) An illustration of scale toleration (red lines represents zoom out and blue lines represent zoom in operations. [11] (b) An illustration of the rotation toleration process. [11]

A similar method is used to account for rotation. For rotation, the image is sampled along rotated lines. An example of this is shown in Figure 9b.

To deal with translation, first the location of the optical disc is determined. If the optical discs are close enough to each other, within predetermined limits, no further translations are applied. Otherwise, the sample image is translated in order for the optical discs to be in the same location.

Finally, the similarity measure is calculated. The total number of matching vessels between the image and the reference image are counted. The total number is determined by counting intersections of vessels for all reference lines. From this a normalized similarity measure is calculated.

## 4 Comparison

In section 2 we examined each biometrics field. This general overview gives us the ability to compare the characteristics that were defined in section 1. In section 3 we went into a state-of-the-art method in each of these fields. Each of these methods gives different ways of improved performance compared to older methods. This information gives us the ability to compare the Performance characteristic of each method.

### 4.1 Characteristics evaluation

An empirical comparison of the characteristics given in section 1 is presented in table 1. In the following subsections we will expand on each of these and corroborate this information with our own findings from the literature.

| Traits | Fingerprint | Palmprint | Retinal Vascular |
|---|---|---|---|
| Universality | Medium | Medium | High |
| Uniqueness | High | Medium | High |
| Permanence | Medium | Medium | High/Medium |
| Collectability | Medium | High | Medium/Low |
| Acceptability | High | Medium | Low |
| Circumvention | Low | Medium | High |

Table 1: Empirical evaluation of different biometrics based on their characteristics. [15]

#### 4.1.1 Universality

Each of the biometric traits are universal for nearly every person. Barring serious injury or disability every person should have all three of these biometrics.

#### 4.1.2 Uniqueness

As stated in section 2, retinal vascular networks of different people, even those of identical twins, are vastly different [17]. The same also holds for fingerprint [6] and palmprint [10] identification. Palmprint does have issues with uniqueness based on low-resolution images, that only distinctively show principal lines and some wrinkles.

#### 4.1.3 Permanence

For all three biometric traits there is at least some measure of permanence over a persons lifetime. Fingerprints might however be damaged due to medication or disease [12]. Retinal vasculature has a similar issue with degradation due to disease or old age, as stated in section 2.3. For palmprints the low-resolution features probably degrade the least over a persons lifetime. This is however not completely clear from the literature.

#### 4.1.4 Collectability

Given the fact that fingerprints can be extracted using a simple, cheap camera. The collectability is very high. This is evident from the fact that such technologies are included in the unlock buttons in smart phones. The low-resolution features of the palmprint identification method of section 3.2 require only low-grade imaging technologies. For retinal vascular imaging the process is more complex, because a specialized fundus camera is needed, which are very costly compared to cameras used for the other two methods.

#### 4.1.5 Acceptability

Given the fact that fingerprint identification is implemented in nearly every flagship smartphone nowadays and that they are still widely used among the population we expect that acceptability is high. With respect to forensic science the practice is widely used, the people whose fingerprints are extracted for this purpose however might be less accepting. People will probably accept palmprint identification as an identification method in a similar way to the way they do fingerprints because it regards an even more generic part of the hand, but given the fact that the adoption is not as high, this still remains to be seen. The way retinal imaging is performed is pretty intimate, with the person having to get close up to the camera, and people might find taking an image of the back of their eye pretty invasive. The literature however is not very conclusive on this point, so the best we can do is speculate. In places where extremely high security is warranted people probably take such measures for granted.

#### 4.1.6 Circumvention

With regards to circumvention retinal vasculature is the most secure method, given the fact that retinal vasculature can not be altered through plastic surgery or other methods, which is possible for fingerprints and palmprints [11].

### 4.2 Performance evaluation

Now we have come to the more interesting part with respect to the technical side of the issue. How does each state-of-the-art method perform and how do they stack up to their predecessors and each other. An overview of some performance results is displayed in table 2.

#### 4.2.1 Storage

The fingerprint method is specifically optimized for matching examples over a large dataset. The storage capacity of the fingerprint identification is lower than the retinal vascular and palmprint methods. This optimization is achieved by not storing a grayscale image, but by only storing the unique identifying features of the image. The retinal vascular and palmprint methods are very similar in that they save binary images. The palmprint method is slightly more efficient however, since it discards a large portion of the data of the original image.

#### 4.2.2 Accuracy

The accuracy of the algorithms is presented as the false acceptance rate (FAR). The FAR is defined as the percentage of invalid inputs which are incorrectly accepted. The accuracy of the algorithm are calculated over a corrupted and normal (uncorrupted) dataset. The corrupted dataset has extra noise added to simulate failures that often occur on cameras.

The palmprint and retinal vascular methods both achieve an accuracy of 100% on the uncorrupted dataset. The results of the retinal vascular and palmprint identification are less reliable than the fingerprint example, since both use a limited dataset of less than 1000 persons.

#### 4.2.3 Speed

The matching and verification speed is obtained of each biometric method. The matching speed is obtained as the amount of seconds it takes to compare an example. The matching speed metric is important for identification of an example in a large dataset. Fingerprint identification has the fastest matching speed. It can achieve nine million fingerprint comparison per second on a single gpu, since it only uses bit-wise operators. The matching speed of the retinal vascular method was not clearly defined, however it is implied in the text that it is reduced to several milliseconds. It is also claimed that the results are quite promising for data mining on medical databases [11].

The verification speed is retrieved as the amount of time it takes for one example to be verified. The verification process includes preprocessing, feature extraction, and matching. The verification of a fingerprint can be done real-time without the user perceiving latency. The retinal vascular identification performs very slow, since a lot of operations are used in the segmenting of the image.

The speed of palmprint identification is near real-time for both matching and verification speed.

| Biometric | Storage(bytes) | Accuracy(uncorrupted) | Accuracy(corrupted) | Speed(matching) | Speed(verification) |
|---|---|---|---|---|---|
| Fingerprint | 1.068 | 98.1% | - | $1.11 \times 10^{-7}$s | 0.0042s |
| Palmprint | 5.000 | 100% | 98.6% | 0.295s | 0.72s |
| Retinal vascular | 54.150 | 100% | 97% | - | 11.67s |

Table 2: The results of the biometric analysis techniques

## 5 Discussion

In general, we can say that fingerprint and palmprint identification are more suitable for low-critical applications, that require less intrusiveness. Retinal vascular identification is very useful for high-critical application, that are less dependent on their intrusiveness.

The comparison shows that each of these biometric traits and methods has its own advantages and drawbacks. In terms of large scale applications fingerprint methods are probably the most useful given the efficiency of the minutia-based method.

Where high security is warranted retinal vasculature is probably the most secure. A large benefit of the proposed method is its resilience to image rotation, scale, and translation, and the speed of the matching algorithm which makes it a better competitor to the other methods, than previous retinal vascular methods. A main drawback of retinal is the high cost of imaging technology that is required. Another issue is the 11.67 second verification time, which can be prohibitive in situations where speed is required.

Based on the implementation of the methods and the general advantages and disadvantages we can say that the minutia-based cylinder-code fingerprint analysis is a very promising method with regards to efficient storage and matching in cases where large amounts of biometric data have to be compared.

The palmprint method has an advantage with respect to low-resolution images and collectability from an individual. This biometric method could be incorporated into cheap phones to provide accurate biometric identification with a simple camera. The use of palmprint biometrics could become more common if it is used for small cheap devices with a camera, which are common in the so-called Internet of Things.

## 6 Conclusion

We have shown fingerprint, palmprint and retinal vascular biometric analysis methods to identify a person. The corresponding state-of-the-art methods were outlined for each biometric analysis method. These biometric methods have been evaluated based on their charateristics. Given the fact that each described method uses a different field of biometric traits (i.e. fingerprint, palmprint, retinal vasculature), it is hard to make a direct numerical comparison between the algorithms. However, by using the more abstract charateristics we have tried to distill the advantages and disadvantages of each technique in comparison to each other and other methods within their field. In this way the results give insight into the functional use-cases of each biometric method. While also giving a look into the most innovative aspects of each method. By doing so it might be possible to gain insight into the way each method might be usable in real-world applications.

By giving insight into which aspects of each method are most innovative and creative we hope to inspire further research into these concepts. Future research could possibly be done into incorporating the minutia-based bit-wise structure of MCC into palmprint and retinal vascular comparison methods.

Köse et al. [11] claims that retinal vascular cameras can become more common in the near future, since the costs of medical image technology is decreasing and the amount of research is increasing. Therefore, the use-cases of fundus cameras might expand and we might even see fundus cameras being integrated in handheld wearables. Future research could be done to create faster verification algorithm to support this trend for new wearable sensors.

## References

[1] J. H. Arndt. Optical alignment system, May 8 1990. US Patent 4,923,297.

[2] F. Blanchini and S. Miani. *Set-theoretic methods in control*. Springer, 2008.

[3] R. Cappelli, M. Ferrara, and D. Maio. A fast and accurate palmprint recognition system based on minutiae. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 42(3):956–962, 2012.

[4] R. Cappelli, M. Ferrara, and D. Maltoni. Minutia cylinder-code: A new representation and matching technique for fingerprint recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(12):2128–2141, 2010.

[5] J. Feng and A. K. Jain. Fingerprint reconstruction: from minutiae to phase. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(2):209–223, 2011.

[6] Y. Han, C. Ryu, J. Moon, H. Kim, and H. Choi. A study on evaluating the uniqueness of fingerprints using statistical analysis. In *International Conference on Information Security and Cryptology*, pages 467–477. Springer, 2004.

[7] R. B. Hill. Apparatus and method for identifying individuals through their retinal vasculature patterns, Aug. 22 1978. US Patent 4,109,237.

[8] R. B. Hill. Rotating beam ocular identification apparatus and method, July 12 1983. US Patent 4,393,366.

[9] A. K. Jain, P. Flynn, and A. A. Ross. *Handbook of biometrics*. Springer Science & Business Media, 2007.

[10] A. Kong, D. Zhang, and G. Lu. A study of identical twins palmprints for personal authentication. In *International Conference on Biometrics*, pages 668–674. Springer, 2006.

[11] C. Köse, C. İkibaş, et al. A personal identification system using retinal vasculature in retinal fundus images. *Expert Systems with Applications*, 38(11):13670–13681, 2011.

[12] M. M. Lowe. A comprehensive study of the effects of chemotherapy on friction ridge detail. 2016.

[13] D. Maltoni, R. Cappelli, and D. Meuwly. Automated fingerprint identification systems: From fingerprints to fingermarks. In *Handbook of Biometrics for Forensic Science*, pages 37–61. Springer, 2017.

[14] D. Maltoni, D. Maio, A. K. Jain, and S. Prabhakar. Handbook of fingerprint recognition. In *Springer Professional Computing*, 2003.

[15] W. Meng, D. S. Wong, S. Furnell, and J. Zhou. Surveying the development of biometric user authentication on mobile phones. *IEEE Communications Surveys & Tutorials*, 17(3):1268–1293, 2015.

[16] D. Palma, P. L. Montessoro, G. Giordano, and F. Blanchini. Biometric palmprint verification: A dynamical system approach. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 2017.

[17] P. Tower. The fundus oculi in monozygotic twins: report of six pairs of identical twins. *AMA archives of ophthalmology*, 54(2):225–239, 1955.

[18] M. L. Vijilious and V. S. Bharathi. Texture feature extraction approach to palmprint using nonsubsampled contourlet transform and orthogonalmoments. *International Journal of Future Computer and Communication*, 1(3):298, 2012.

[19] X. Wu, D. Zhang, K. Wang, and B. Huang. Palmprint classification using principal lines. *Pattern recognition*, 37(10):1987–1998, 2004.

[20] S. Yamamoto. Image processing and automatic diagnosis of color fundus photographs. *Proc. ICPR, 1974. 8*, pages 268–269, 1974.

[21] H. Yokouchi, S. Yamamoto, T. Suzuki, M. Matsui, and K. Kato. Automatic pattern recognition of fundus photography. i. pattern recognition of crossing phenomena of fundus photography by extraction of contour lines of blood vessels. *Iyo denshi to seitai kogaku. Japanese journal of medical electronics and biological engineering*, 12(3):123–130, 1974.

# A comparison of ensemble methods: AdaBoost and random forests

Emilio Oldenziel and Erik Bijl

**Abstract**—In recent years multiple approaches for training a classifier on data have been proposed. One approach is to combine several weak and simple classifiers in a so-called ensemble. Two well-known implementations of ensemble methods are AdaBoost and random forests. The algorithms of each of these methods are well-known but explaining their behaviour is still an active field of research.

In this contribution, a comparison of the behaviour of AdaBoost and random forests is performed. The authors expand on the findings by Wyner et al. (2015) who characterise these classifiers as "spiked-smooth classifiers". A spiked-smooth classifier satisfies two properties: interpolation and self-averaging. In the experiments described in this paper we test the interpolation effects of AdaBoost and random forests by adding multiple levels of label noise. We performed our experiments using two existing data sets to find an answer to our research question: what are the similarities and differences in behaviour between AdaBoost and random forests?

Our main findings include that AdaBoost and random forests are more robust to noise than non-interpolating classifiers. In addition we found that random forests-based ensemble methods are generally more robust to noise than AdaBoost-based methods.

**Index Terms**—Machine learning, Classification, Ensemble methods, AdaBoost, Random forests, Interpolation, Self-averaging.

## 1 Introduction

In supervised machine learning a classifier is trained by sample input/output-pairs. The process is divided into two phases: a training phase and a test phase. During the training phase sample inputs and outputs are presented to the classifier. From these samples the classifier predicts the output with respect to the novel data. These predictions are performed during the test phase, which allows one to determine a measure of performance. Often the test error rate, the percentage of wrong predictions, is used as a measure of the classifiers performance. The goal in supervised learning is to minimise the test error rate. The lower the test error rate of a classifier, the more successful the classifier will correctly assign a classification to unknown novel samples.

In recent years many approaches have been proposed for classifier training. One approach employs ensemble or voting methods and combines several independent classifiers into one final classifier. The final classifier is constructed by a vote of the independent classifiers trained on the data. Two well-known families exist within ensemble methods which are Boosting and Bagging (also known as Bootstrap aggregating).

Boosting is an ensemble method that iteratively trains weak learners on different subsets of the data set. In each round a sample's contribution is changed relatively to the correctness of the classifier trained in the previous round. For incorrectly classified data the weights are increased and for correctly classified data they are decreased. The training algorithm focuses on the large weights which results in a correct classification of all samples. Bagging is an ensemble method that creates subsets in a different manner. It trains independent classifiers on data sets sampled from the original data set. The used sampling method is sampling with replacement: meaning that one single sample can occur multiple times in the constructed subset.

Following the conceptual introduction of both methods multiple implementations have been suggested. This paper discusses two well-known implementations. The first implementation is a boosting method called adaptive boosting, better known as AdaBoost (Freund and Schapire, 1996). The second implementation is a bagging method called random forests which was introduced after AdaBoost by Leo Breiman. (Breiman, 2001).

Although the algorithms of AdaBoost and random forests are well-known, explanations of the behaviour of both methods vary and are currently an active field of research (Schapire, 2013) (Cao et al., 2013) (Mukherjee et al., 2013). It has been argued that these algorithms in certain aspects are similar. Already in 2001 a connection was noticed between random forests and AdaBoost (Breiman, 2001). Breiman made a conjecture hypothesising that AdaBoost is a random forest of forests. A more recent publication (Wyner et al., 2015) characterised the behaviour of both AdaBoost and random forests as "spiked-smooth" algorithms. Wyner et al. (2015) performed multiple experiments that explain certain behaviour of AdaBoost and random forests. In one of the experiments the robustness of AdaBoost and random forests was tested by adding label noise to the data set.

The main focus of this paper is to compare the algorithms of AdaBoost and random forests by expanding on the experiments of Wyner et al. (2015). Wyner's experiments visualise how interpolation can provide robustness in a noisy setting. This is achieved by training AdaBoost and random forests classifiers using noisy data sets. The research question we want to answer is: what are the similarities and differences in behaviour between AdaBoost and random forests? In our experiments a larger amount of noise than in the experiments by Wyner et al. (2015) is added to the data set in order to attempt to highlight differences between AdaBoost and random forests. Our results indicate differences in behaviour of AdaBoost and random forests. These differences were not discovered in previous research and are a new contribution to the understanding of behaviour of AdaBoost and random forests.

In Section 2 we provide some background information on the current state of knowledge in AdaBoost and random forests. We start with discussing the theory underlying ensemble methods and in particular AdaBoost and random forests. Then certain theories are discussed that focus on the similarities of both algorithms. These include Breiman's connection between the algorithms and findings of Wyner et al. (2015) characterising both algorithms as spiked-smooth classifiers. In Section 3 the set-up of our experiments is discussed. Section 4 discusses two real-life data sets used in our experiments. We summarise the findings of our experiments in Section 5 and discuss these in Section 6. A brief conclusion on our findings is provided in Section 7.

## 2 Background information

In this paper two ensemble methods are specifically examined. We initially provide a general overview of ensemble methods.

### 2.1 Ensemble methods

Several weak classifiers, only slightly better than random guessing can be combined to form one more powerful classifier. This approach is called voting, committee or ensemble method. Different weak classifiers can be chosen to form an ensemble. Decision trees are most com-

- *Emilio Oldenziel is a first year Computing Science master student at the University of Groningen .*
- *Erik Bijl is a first year Computing Science master student at the University of Groningen.*

**Algorithm 1** AdaBoost (Schapire and Freund, 2012)

```
Given $(x_1,y_1),\dots,(x_m,y_m)$ where $x_i \in X, y_i \in \{-1,+1\}$.
Initialize: $D_1(i) = 1/m$ for $i = 1,\dots,m$.
For $t = 1,\dots,T$:
    • Train a weak learner using distribution $D_t$.
    • Get weak hypothesis $h_t : X \to \{-1,+1\}$.
    • Aim: select $h_t$ to minimise the weighted error:
          $\varepsilon_t = \mathbf{Pr}_{i \sim D_t}[h_t(x_i) \neq y_i]$.
    • Choose $\alpha_t = \frac{1}{2}\ln(\frac{1-\varepsilon_t}{\varepsilon_t})$.
    • Update, for $i = 1,\dots,m$:
          $D_{t+1}(i) = \frac{D_t(i)}{Z_t} \times \begin{cases} e^{-\alpha t} & \text{if } h_t(x_i) = y_i \\ e^{\alpha t} & \text{if } h_t(x_i) \neq y_i \end{cases}$
          $= \frac{D_t(i)\exp(-\alpha_t y_i h_t(x_i))}{Z_t}$
      where $Z_t$ is a normalisation factor (chosen so that $D_{t+1}$ will be a
      distribution).

Output the final hypothesis:
          $H(x) = sign(\sum_{t=1}^{T} \alpha_t h_t(x))$.
```

monly used as learning algorithms in ensembles (Hong et al., 2018). To enforce variation between the weak learners, they must be trained on different subsets of the actual data. Creating these different subsets of the data is the main task for an ensemble method. The realisation of these subsets is the main difference between AdaBoost and random forests. Both algorithms are discussed in more detail in the following Sections.

## 2.2 AdaBoost

AdaBoost is a realisation of the general method Boosting originally invented by Freund and Schapire (1996). In boosting a given weak learning algorithm is repeatedly trained on different parts of the used data set. The algorithm of AdaBoost is formally described in pseudocode in Algorithm 1.

The algorithm expects training samples $(x_1,y_1),\dots,(x_m,y_m)$ where $x_i$ is a feature vector from $X$. The labels of each sample are given by the value $y_i \in \{-1,+1\}$. Note that for simplicity a binary classification is considered, unless explicitly stated otherwise.

The algorithm performs $T$ so-called boosting rounds where in each boosting round the same weak learner is trained on a distribution of the data set. To achieve different classifications of the data by the same learning algorithm these distributions must vary. As Freund and Schaphire argue: "the key idea behind boosting is to choose training sets for the base learner in such a fashion as to force it to infer something new about the data each time it is called". Therefore AdaBoost needs a different distribution $D$ over the training samples. During boosting round $t$ a distribution $D$ is used and denoted as $D_t$. The amount of samples in the data set is denoted as $m$ and the corresponding weights of the distribution $D_t(i)$ are normalised to $1/m$.

The algorithm proceeds by training the weak learner repeatedly in a series of boosting rounds. Implementations of such a weak learner can vary but AdaBoost with trees is mostly used. Breiman (NIPS Workshop, 1996) referred to AdaBoost with trees as the "best off-the-shelf classifier in the world". The weak learner yields a hypothesis of each sample to a label $\{-1,+1\}$. This hypothesis is evaluated by an error function which can be seen as an indicator of the performance of the weak classifier. The weights represent the value of relevance of a hypothesis and depend on this error. A classifier that performs well, yields a low error and results in a high weight $\alpha$. The last step in each round is to update the distribution. The updated distribution will be given to the weak learner in the following round. The weights are increased by a factor $e^{\alpha t}$ when a sample is wrongly classified and by a factor $e^{-\alpha t}$ for a correctly classified sample.

After performing $T$ boosting rounds the sum over all hypotheses derived in each round multiplied with the weights of those rounds is computed. The final hypothesis is the result of a sign function on this sum and can be seen as a simple majority vote.

## 2.3 Random forests

Random forests (Breiman, 2001) is an ensemble method that uses the Bagging approach of machine learning. Random forests creates a large collection of uncorrelated trees from different distributions $X^*$ of the data set and averages the output of these trees.

The pseudocode algorithm of random forests is presented in algorithm 2. The algorithm consists of $B$ rounds in the training phase. In each round a distribution, also known as bootstrap sample, $X^*$ of size $N$ is taken from the training data. A bootstrap sample is constructed by sampling with replacement of $N$ (possibly similar) samples from the training data. On $X^*$ a decision tree is created by the steps given in the algorithm. The resulting tree is a deep tree because each node has a minimum size set by the minimum node size $n_{min}$.

During the training phase in each round $b$ a tree $\{T_b\}$ is computed. After the training phase the resulting classifier is a majority vote of all these $B$ trees.

**Algorithm 2** random forests (Hastie et al., 2001)

```
1. For $b = 1$ to $B$:
    (a) Draw a bootstrap sample $X^*$ of size $N$ from the training data
    (b) Grow a decision tree $T_b$ to the data of $X^*$ by doing the following
        recursively until the minimum node size $n_{min}$ is reached:
            i. Select randomly $m$ of the $p$ variables
           ii. Pick the best variable/split-point from the $m$ variables and
               partition
2. Output the ensemble $\{T_b\}_b^B$

Let $C_b$ be predicted class of tree $T_b$, then the final classifier is
$C_B$ =sign($\sum_{b=1}^{B}\{C_b\}$).
```

## 2.4 Breiman's conjecture

In addition to the introduction of random forests, Breiman included a brief comparison to AdaBoost. In this comparison, Breiman formulated a conjecture stating that AdaBoost is equivalent to random forests under certain conditions. This is the case when AdaBoost randomly selects the weights on a training set from a distribution $Q_\pi$. The probability $Q_\pi$ is the proportional probability distribution of the weights $w(k)$. Breiman showed that after a number of runs, the resulting distribution of AdaBoost is defined by:

$$Q_\pi = log[(1-error(k))/error(k)] \tag{1}$$

where $error(k)$ is the weighted training set error. The conjecture holds when the operation $f(\phi)$, with $f$ as the function defined by the weight space and $\phi$ by the base classifier, is ergodic to the invariant function $\pi(dw)$. In other words, the selection of weights by AdaBoost becomes random and similar behaviour as random forests is obtained. As pointed out by Belanich and Ortiz (2012), Breiman's conjecture was based on the assumption of having an infinitely large data set. This assumption is not a condition that can be simulated by performing experiments since infinitely large data sets are not available. Breiman also stated this by mentioning that there was a fundamental flaw in trying to transfer the results to finite data sets (Belanich and Ortiz, 2012). Therefore we can conclude that Breiman's conjecture is not suited for an actual comparison of AdaBoost and Random forest.

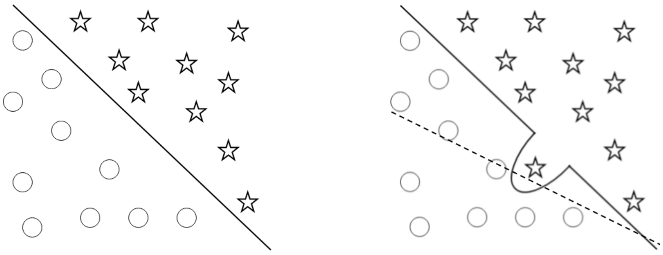
## 2.5 Spiked-smooth classifiers

A recent publication (Wyner et al., 2015) showed that current explanations of the success of AdaBoost and random forests are incomplete.

Instead, a new perspective on both algorithms was proposed. Both algorithms, AdaBoost and random forests train a “spiked-smooth” classifier. A spiked-smooth classifier satisfies two properties: self-averaging and interpolation. It was shown that these properties hold for both AdaBoost and random forests. By showing this, a different light is shed on the similarity between the algorithms. The properties of spiked-smooth classifiers hold for AdaBoost and random forests and are described below.

### 2.5.1 Interpolation

Using the term interpolation can be misleading in the context of Computing Science because it is used differently depending on the author or context. Here an interpolating classifier is defined as a classifier that perfectly fits the training data: meaning that for each sample within the training data the classifier gives the correct label in its output. As a result, the classification error on the training data is zero after the training phase. In most cases zero training error implies that the classifier overfits the training data and does not perform well in the test phase due to bad generalisation. Wyner et al. (2015) reverse this notion by demonstrating that interpolation can prevent overfitting. In order to prevent overfitting, interpolation must only be applied locally. Locally applied interpolation does not corrupt the vote because only the area close to the sample is affected. In this way noisy samples that deviate from existing samples are effectively smoothed out. This principle is illustrated in Figure 1. In Figure 1a a simple decision boundary between stars and circles is shown. In Figure 1b a noisy sample that deviates from the existing samples is added. This noisy sample is the star in the centre of the image. The figure illustrates the difference between an interpolating classifier and a non-interpolating classifier for example logistic regression. The interpolating classifier shown as the solid line is resistant to noise because it interpolates the noisy sample. This results in a perfect fit of the data. The decision boundary given by the non-interpolating classifier denoted by the dotted line is shifted by the noisy sample. As a result the non-interpolating classifier does not perfectly fit the data. From this it can be understood that an interpolating classifier is able to adapt locally to the noisy point. The overall fit does not become corrupted by adding a noisy sample.



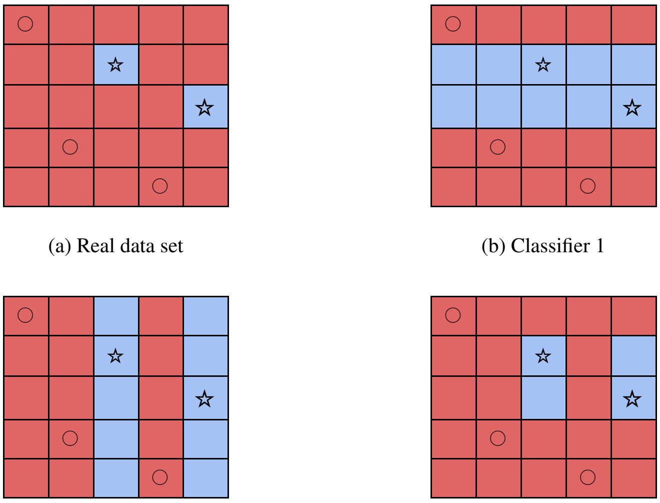
(a) Simple decision boundary

(b) Addition of noisy sample

Fig. 1: Figure (a) shows a simple decision boundary. Figure (b) illustrates the effect of adding a noisy sample on an interpolating classifier (solid line) and a classifier such as logistic regression (dashed line).

### 2.5.2 Self-averaging

The second property of spiked-smooth algorithms is self-averaging. This property can be seen as smoothing out all classifiers of the ensemble methods. Self-averaging combines all interpolating classifiers to one global classifier by averaging them. An illustration of this property is shown in Figure 2. A real data set of stars and circles is given in Figure 2a. Two classifiers are trained on this data, resulting in predictions of novel data shown by Figures 2b and 2c. The self-averaging property is shown in Figure 2d as an average of the classifiers. It is easy to see how random forests satisfies self-averaging: it is the average over all independent trees. For AdaBoost this is harder to demonstrate because it iteratively trains a classifier. An interesting observation is made in Wyner et al. (2015), when they decompose the classifier during the rounds of AdaBoost. Each 100 rounds a new classifier is trained on the weights carried over from the previous classifier. All 10 resulting classifiers are classifiers that reduce the training error. The final classifier is an interpolating classifier by taking the weighted average over these 10 classifiers. Given this observation it can be concluded that boosting in this way is a self-averaging process.



(a) Real data set

(b) Classifier 1

(c) Classifier 2

(d) Averaging classifier

Fig. 2: The labels of a real data set shown in 2a are predicted by two classifiers shown in 2b and 2c. The average of these two classifier is shown in 2d.

## 3 Methods

To validate and expand the findings of Wyner et al. (2015) two of the real data experiments are re-visited. To obtain differences between the behaviours of AdaBoost and random forests the robustness of the spiked-smooth principle is tested. This can be done by analysing the effects of the error rates while adding more noise to the data sets. Doing this will provide an indication of the interpolation properties of both algorithms. As a reference one nearest neighbour classifier (1-NN) is considered. By definition a nearest neighbour classifier has no interpolating properties. Therefore, we use this classifier to set a reference score that can be compared with the scores resulting from the interpolating classifiers. The difference between these scores indicates the effect of interpolating properties on the data set.

### 3.1 Classifiers

The experiments were performed using decision trees as a weak learner. By definition this is the case for the random forests. The classifier produced by AdaBoost can use decision trees as its basic classifier. The implementations of the experiment were written using the scikit-learn[1] toolkit and can be found on Github[2]. The reading of the data sets into the experiments is performed using Pandas, which is a Python framework for handling data sets[3], the samples are stored in a Pandas DataFrame.

### 3.2 Training parameters

The parameters in the training phase are delected according to the settings by Wyner et al. (2015).

- $\lambda_{noise}$: amount of label noise that is added to the data set performed by inversion of the labels. The interval is $0 \leq \lambda_{noise} \leq 1$.

[1] scikit-learn: `http://scikit-learn.org`

[2] https://github.com/EmilioOldenziel/StudentColloquium

[3] Pandas: `https://pandas.pydata.org/`

- Maximum tree depth: number of nodes that the tree maximally has in depth. For AdaBoost we set this number to 8 and for random forests to infinity (no maximum) as in Wyner et al. (2015).

- $T$ : number of boosting rounds in AdaBoost. In each round the error is used to produce weights $w$ for the next boosting round. In our experiments we set $T$ to 100, which is based on the results from Wyner et al. (2015) with tree depth 8 where the test error displays a minor decrease after 100 rounds.

- $B$: amount of random forests that are grown. In each round a random forests is created which makes $B$ equal to the amount of boosting rounds $T$.

- $R_{validation}$: amount of validation splits with test- and trainingset. We set $R_{validation} = 10$ so that we get a well represented average but no time consuming training phase.

### 3.3 Validation

Both classifiers are trained and validated $R_{validation}$ times. In each validation split $r < R_{validation}$ a classifier is initialised with the parameters above. The randomised split of the data set is 70/30, making 70% of the samples part of a training set and the remaining 30% of a test set. The training phase consists of $T$ and $B$ boosting rounds for AdaBoost and random forests, respectively. In each round the classifier is validated using the training and test set. The validation of the training- and test set at boosting round $t$ and $b$ are stored in $R_{validation} * T$ and $R_{validation} * B$ matrices. When classification is completed, the results are averaged over all validation rounds $R_{validation}$. This approach is the same as performed by Wyner et al. (2015) to reproduce their validation scheme. The results are displayed in plots of average error rates per boosting round.

### 3.4 Additional Noise

To test the robustness to noise of AdaBoost and random forests we add noise to the labels of the data sets. The addition of noise to the data set is performed by inverting a corresponding amount of classification labels or so called bit flipping. The amount of label flips is defined by:

$$N_{flips} = \textbf{round}(N_{dataset} * \lambda_{noise}). \qquad (2)$$

Five stages of noise are included in our experiment. The first level includes no noise meaning that the classifiers are trained on the original data sets. The second level is 2% which is an equal amount of noise as in the experiments of Wyner et al. (2015). The other levels of noise are: 5%, 10% and 20% (only for the breast cancer data set, see Section 4.2). We do not expect the classifiers to produce a decent generalisation above of noise above 20%.

## 4 Data

In our experiments two real-life data sets are used. The data sets are described in the following Sections.

### 4.1 Phoneme data set

The UCL Phoneme data set contains 5404 samples of the spoken vowels “nasal” and “oral”. The features consist of the first 5 harmonic frequencies, which are collected by cochlea spectra. This data set is used by Wyner et al. (2015) because of the small number of real valued covariates. The labels have two values: 0 for the vowel “nasal” and 1 for the vowel “oral”.

### 4.2 Breast cancer data set

The second data set that we use is the UCL Wisconsin Breast Cancer data set. This data set is used in several AdaBoost and random forests experiments (Breiman, 2001; Belanich and Ortiz, 2012; Wyner et al., 2015). It contains 569 samples which are considerably fewer samples than the Phoneme data set. Each sample in this data set has 30 positive real valued features. The samples are labelled according to tumor dignity: class label 0 denotes benign and label 1 denotes malignant.

## 5 Results

The experiments result in plots from the validation process, which are captured in a graph per noise level. The figures include test- and training errors for AdaBoost and random forests for 100 rounds denoted as boosting iterations. The test errors after these 100 iterations are also included in Table 1 and Table 2. In these table 1-NN is added as a reference to compare the increase in test score between the interpolating and non-interpolating classifiers.

### 5.1 Phoneme

The test errors on the Phoneme data set for each noise level are shown in Figure 3, the average test error after the training phase is shown in Table 1. For the 10% noise level we increased the number of boosting iterations from 100 to 250 to let the training phases of AdaBoost and random forests fully converge.

| Noise level $\lambda$ | AdaBoost | Random forests | 1-NN |
|---|---|---|---|
| 0.00 | 0.09 | 0.10 | 0.10 |
| 0.02 | 0.12 | 0.12 | 0.13 |
| 0.05 | 0.15 | 0.14 | 0.17 |
| 0.10 | 0.21 | 0.19 | 0.24 |

Table 1: Average test error after 100 boosting rounds with noise added to the phoneme data set ($t = 100$, $b = 100$) for AdaBoost, random forests and 1-NN.

### 5.2 Breast cancer

The results of the breast cancer data set show the same effects in the training phase as with the Phoneme data set, therefore we only included the result of the addition of 5% noise as performed by Wyner et al. (2015). Table 2 shows the results of the validation with the test set for AdaBoost, random forests and 1-NN.

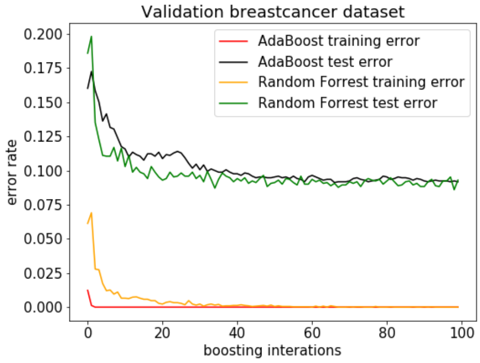


Fig. 4: Breast cancer data set 5% noise ($\lambda = 0.05$) over 100 boosting iterations.

| Noise level $\lambda$ | AdaBoost | Random forests | 1-NN |
|---|---|---|---|
| 0.00 | 0.00 | 0.00 | 0.08 |
| 0.02 | 0.05 | 0.06 | 0.10 |
| 0.05 | 0.07 | 0.08 | 0.19 |
| 0.10 | 0.14 | 0.14 | 0.24 |
| 0.20 | 0.23 | 0.22 | 0.36 |

Table 2: Average test error after 100 boosting rounds with noise added to the breast cancer data set ($t = 100$, $b = 100$) for AdaBoost, random forests and 1-NN.

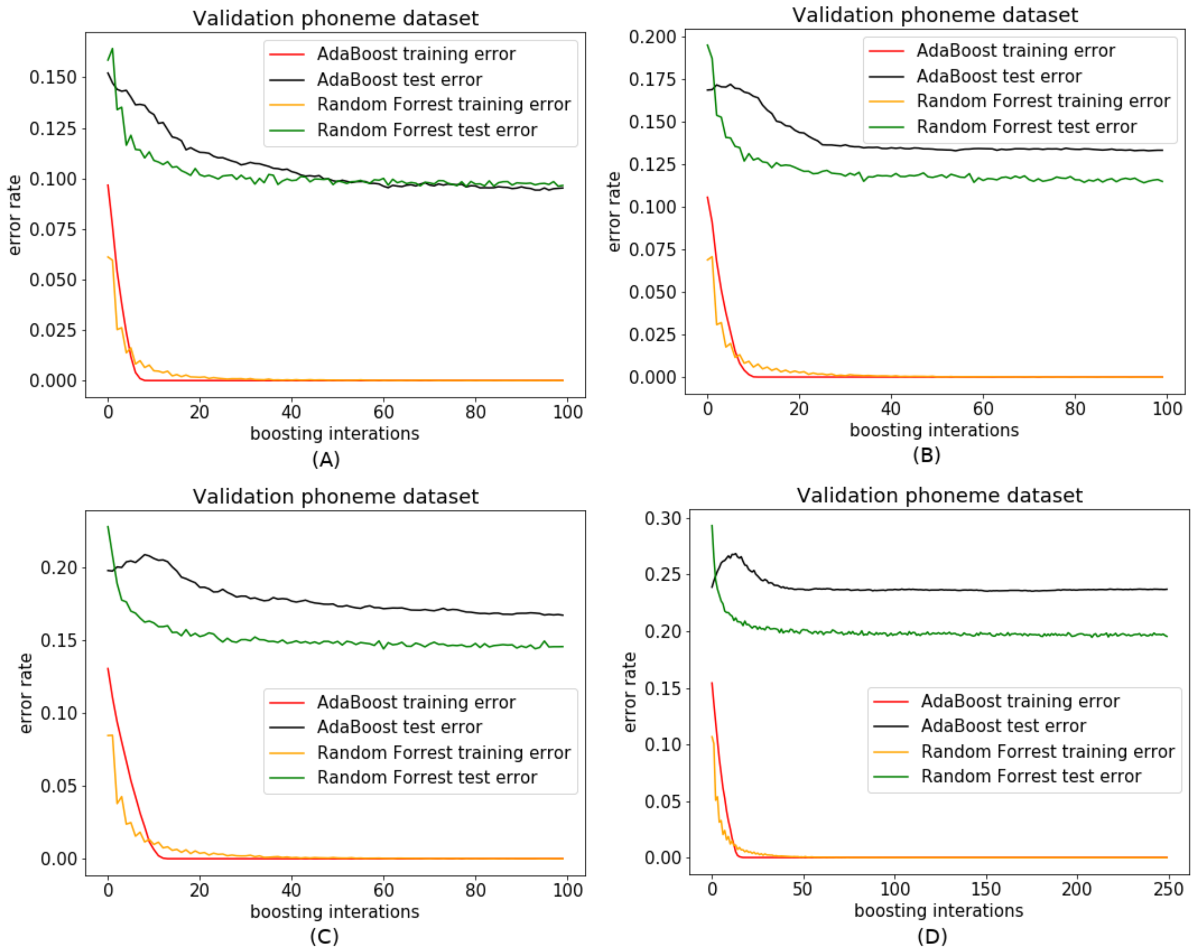


Fig. 3: Validation results for AdaBoost and random forests. A: no noise ($\lambda_{noise} = 0.0$), B: 2% noise ($\lambda_{noise} = 0.02$), C: 5% noise ($\lambda_{noise} = 0.05$) D: 10% noise ($\lambda_{noise} = 0.10$).

## 6 Discussion

The results of our experiment show expected but also some unexpected behaviours during the training phases. As we have described, interpolating classifiers must be more robust to noise than non-interpolating classifiers. It is also expected that performances of AdaBoost and random forests show similar behaviour under all conditions.

Looking at the behaviour of the final classifier given by AdaBoost, one can see that it interpolates the training data. This can be concluded from the error in the training phase without additional noise in Figure 3(A). In each boosting iteration the training error decreases and becomes zero after a small number of iterations. At this point a perfect fit of the training data is obtained and interpolation which is one of the properties for a spiked-smooth classifiers is satisfied. Interpolation does not occur for the test data at this point. In fact we can observe a decreasing linear trend on the test data only after the training data has been interpolated. It seems that AdaBoost first needs to fit each sample in the training data before it can generalise to the samples in the data set. After AdaBoost interpolated the training data we can observe the effect of the second property of a spiked smooth classifier which is self-averaging. The interpolated samples must be smoothed to allow the interpolation to become more local. This is performed by continuing the algorithm even after interpolation of the training data. Because of this continuation, the amount of weak classifiers over which an average is taken increases. Increasing the amount of classifiers weakens the influence of each individual weak classifier. As a result of this phenomenon, interpolation becomes more local and decreases the test error.

The behaviour of random forests on the data set is much more intuitive as shown in Figure 3. In each iteration a decision tree is created for a particular distribution of the data set. Looking at the training error every decision tree decreases the training error. The decreasing trend continues until the training error reaches zero and the training data is interpolated. An interesting observation can be made by looking at the test error when interpolation has been completed. We obtain an approximately constant test error after interpolation of the training data. By definition the random forests algorithm is an average of the decision trees. Averaging more and more decision trees on the data does not seem to decrease the test error.

The experiments add label noise to the data set which enables us to test the robustness of interpolating classifiers. The robustness of AdaBoost and random forests is tested after 100 boosting iterations and compared to 1-NN. Table 1 contains the average test error of this experiment on the phoneme data set. It can be observed that the average test error for AdaBoost and random forests are approximately equal. A similar performance is obtained for both algorithms by the increase of noise. An non-interpolating classifier as 1-NN shows the most significant increase when applied to the phonemes data set when label noise is added. The test error increase between each noise level is also significantly higher for 1-NN compared to the interpolating classifiers. Therefore this experiment strengthens the conclusions made by Wyner et al. (2015) that an interpolating classifier is more robust to noise.

The plots also show differences between the training error of AdaBoost and random forests. Adding more noise to the labels has the effect that the error rate of AdaBoost increases relative to the error rates of random forests. The difference between the error rates of AdaBoost and random forests is significant and can be clearly seen in Figure 3(D). Our second experiment trained the three classifiers on the breast cancer data set. Similar to the phoneme data set we observe comparable behavioural results for the classifiers trained by AdaBoost or random forests. Again we obtained the most significant increase in noise for the 1-NN classifier. Also the error rate of AdaBoost has a higher increase than random forests. These two findings are therefore obtained in both our experiments.

In Wyner et al. (2015) a minimum increase in the test error was found by adding 5% noise ($\lambda_{noise} = 0.05$) to the breast cancer data set against no noise. For AdaBoost this was 0.2% and for random forests 0.39%, which is an increase of less than 0.005. We were not able to reproduce these test error increase results, instead our increase in test error shown in Table 2 was 0.07 and 0.08. This results in a difference over a factor of 10 compared to the ones by Wyner et al. (2015).

## 7 Conclusion

This paper compared the behaviours of ensemble methods produced by AdaBoost and random forests. As discussed in Wyner et al. (2015) the success of both methods can be attributed to the fact that they are spiked-smooth algorithms. Spiked-smooth algorithms satisfy two properties: interpolation and self-averaging. These two properties were described and we performed two experiments on two different real-life data sets. These experiments have shown that the behaviour of AdaBoost and random forests is different in certain scenarios. It can be concluded that both algorithms interpolate the training data. A surprising observation is that the test error of random forests remains constant after interpolation. The test error of AdaBoost starts to decrease after interpolation of the training data. Our main finding is that AdaBoost exhibits a higher error rate than random forests when more noise is added to the labels. We conclude that both algorithms follow a similar approach of combining several weak learners into one composite classifier. Both of these algorithms train their weak learners on different distributions of data sets. The composition of these data sets is different which results in different behaviour of both algorithms.

## 8 Future work

As mentioned in the discussion Section we were unable to reproduce the percentage of mean increase on the breast cancer data set. Further research could examine the possible causes to this problem. Furthermore, we noticed that AdaBoost shows a parabolic shape of the test error in early rounds. This parabolic shape becomes more visible by increasing the percentage of label noise and is caused by the interpolation and subsequently self-averaging of the noisy data samples. This phenomenon could reveal more insight in the strength of interpolation and self-averaging in AdaBoost.

## Acknowledgements

The authors wish to thank Michael Biehl for his supervising role.

## References

Belanich, J. and Ortiz, L. E. (2012). On the convergence properties of optimal adaboost. *CoRR*, abs/1212.1108.

Breiman, L. (2001). Random forests. *Machine learning*, 45(1):5–32.

Cao, Y., Miao, Q.-G., Liu, J.-C., and Gao, L. (2013). Advance and prospects of adaboost algorithm. *Acta Automatica Sinica*, 39(6):745 – 758.

Freund, Y. and Schapire, R. E. (1996). Experiments with a new boosting algorithm. In *Icml*, volume 96, pages 148–156. Bari, Italy.

Hastie, T., Tibshirani, R., and Friedman, J. (2001). The elements of statistical learning. 2001. *NY Springer*.

Hong, H., Liu, J., Bui, D. T., Pradhan, B., Acharya, T. D., Pham, B. T., Zhu, A.-X., Chen, W., and Ahmad, B. B. (2018). Landslide susceptibility mapping using j48 decision tree with adaboost, bagging and rotation forest ensembles in the guangchang area (china). *CATENA*, 163:399 – 413.

Mukherjee, I., Rudin, C., and Schapire, R. E. (2013). The rate of convergence of adaboost. *Journal of Machine Learning Research*, 14:2315–2347.

Schapire, R. and Freund, Y. (2012). *Boosting: Foundations and Algorithms*. Adaptive computation and machine learning. MIT Press.

Schapire, R. E. (2013). *Explaining AdaBoost*, pages 37–52. Springer Berlin Heidelberg, Berlin.

Wyner, A. J., Olson, M., Bleich, J., and Mease, D. (2015). Explaining the success of adaboost and random forests as interpolating classifiers. *arXiv preprint arXiv:1504.07676*.

# A review of Models for Estimating the Power Consumption of Systems

Cristian Capisizu and Orest Divintari

**Abstract**—Power consumption has become an increasing concern in computing systems and data centers, which indicates the need for power modeling and estimation for all components of a system. The estimation of power consumption plays a crucial role in increasing the performance of computer systems and reducing the costs of data centers.

This study analyzed four methods for modeling the energy consumption of systems. The stochastic model, a probabilistic approach which employs the statistics of CPU utilization and aims to determine the relationship between the power that is consumed by a processor and the work it accomplishes. The second method uses a thermal sensor on each core of the processor and software-based thermal management techniques. The third method, models the electricity consumption of servers as accumulated layers depending on the software that is running on them. The last model focuses on monitoring processor's performance events for online measurement of complete system power consumption.

The approach that will be used in this paper will firstly analyze the aforementioned methods and compare them in order to establish which method is the most efficient for estimating the power consumption of systems.

**Index Terms**—power consumption, processors, load placement, temperature rising.

## 1 Introduction

Power consumption has become an increasing concern in computing systems and data centers, which points the need for power modeling and estimation for all components of a system.The estimation of power consumption plays a crucial role in increasing the performance of computer systems and reducing the costs of data centers.

Data centers play an important role on global electrical energy consumption. Understanding their power dissipation is a key aspect to achieve energy efficiency which will result to reduced costs. In addition, as new software technologies appear, the demand for more efficient hardware components is growing, thus technology has allowed integration of multiple cores in a single processor. However, the energy that is consumed from each core increases the heat density of processors. As a result, the efficiency of processors is reduced.

The power consumption of systems has been studied extensively in order to improve the performance computing systems, to implement dynamic power management policies, to reduce the costs of data centers and to examine the proportionality between the power that is consumed by processors and the work they accomplish. The discussion in this research paper will focus on servers in data centers, in multi-core processors, and even microprocessors, because the models used for estimating power consumption in each separate case differs significantly.

In the first sections of this study, a thorough analysis of four different models for estimating power consumption is presented.

The aim of this study is to find which of the models, is able to estimate more accurately the electrical energy consumption of processors and computer systems in general.

## 2 Models

### 2.1 Runtime Temperature-Based Power Estimation

While estimating the power consumption of systems, it is essential to take a look at multi-core processors because the new trend in industry proves that the numbers of cores per die are constantly increasing [10]. This increasing trend will lead to high power consumption of single dies with multiple-cores and thus to very high heat density [10]. It is widely known fact that thermal dissipation is proportional to energy consumption. The more power is consumed the more heated the multi-core processors become. So in order to lower the thermal strain on the dies, the power consumption of the processor must be adjusted. A solution for this situation was the adoption of software based management techniques. [6] Adjusting the processor's power consumption can be done through voltage and frequency scaling. The processor's speed control will lead to reduced power. The appropriate frequency and voltage can be calculated for each core in particular and based on the hot-spot temperature sensed by on die sensors. This can be done by using on-die voltage regulator technology. The feedback control mechanism resulting will be used to adjust the processors speed. Even though this solution seems optimal there are however two aspects that have to be taken into consideration : the optimal design should be specific to a particular processor design and the number of instructions fetched for execution can be throttled to control processors speed. A very utilized method of optimizing throughput of multi-core processors is using convex optimization methods. This is done quite remarkably by minimizing degradation of the throughput. The voltage and the frequency are optimized for each core by using convex dynamic optimization [8].However it depends on the number of cores in a die, because it can become overly expensive. It is also very important to have an accurate dynamic power estimation since throughput is often proportional to power consumption rather than temperature. The following equation is used to solve dynamic power estimation:

$$C * dT/dt + S * T = B * g \tag{1}$$

where T is the die temperature; g is the power consumption at each power source; B is the mapping matrix between power sources and mesh cell grid points; C is the heat-capacitance matrix; and R is the thermal-resistance matrix. Eq. (1) describes a thermal RC network in the time domain. As we partition a die into N distinct grid points, we can reconstruct the power consumption of each N grid point with its temperature. When 1) a die is partitioned into a large number of uniform mesh grid points and 2) absolute temperature and its gradient in each grid cell are available, the total power consumption of each core as well as the entire multi-core processor can be estimated using Eq. (1) [4].

- *Cristian Capisizu is a master student at the University of Groningen, E-mail: c.capisizu@student.rug.nl.*
- *Orest Divintari is a master student at the University of Groningen, Inc., E-mail: OrestisDivintari@gmail.com.*

### 2.1.1 Dynamic Power Estimation in a Single Core

In a single core processor, estimating dynamic power consumption is done by assuming that there is only one temperature sensor. However this method has some degree of uncertainty due to the fact that there is an asymmetry in the number of inputs and outputs as presented in Eq. (1). Converting multiple inputs to single outputs might be possible sometimes in this particular situation, but the reconversion of a single output to multiple inputs is not something desired by researches. The reason for that is that any single output can be made by only every single input. The researchers in this situation provided more inputs in the reconversion process that corresponded to more output in the original process. In order to reconstruct both of the power and temperature profiles, the sampling interval should be decided based on the coverage area of a single sensor.

### 2.1.2 Dynamic Power Estimation in Multi-Cores

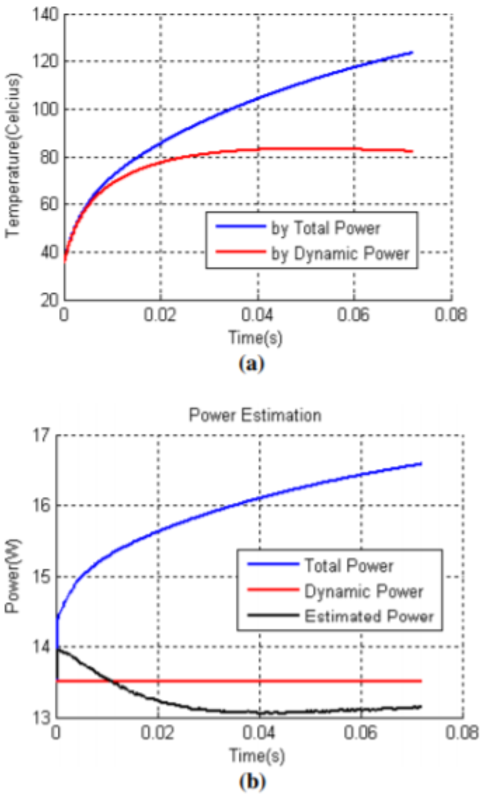


Fig. 1. Dynamic power estimation of a core in a multi-core processor considering leakage power consumption [4].

Dynamic power consumption of each core in multi-core processor can be estimate by utilizing some of the same methods adopted by the dynamic power estimation in a single core processor. Since the thermal management technique has been adopted to optimize processor throughput while thermal constraint is kept, it is also very important to estimate dynamic power consumption of each core because power consumption is directly proportional to the performance of a core [7]. However, if there is a multiple number of active cores in a die, each of them dissipate power and the temperature map of each core is not directly proportional to the power consumption of each core. In Fig. 1 shows the power estimation of a core in a multi-core processor. In a die, there are several other power sources in other cores and their power consumption fluctuates while the power value of a core is constant.

### 2.1.3 Application of Dynamically Estimated Power for Performance Optimization

The throughput of a many-core processor can be optimized with a thermal constraint. The maximum throughput can be achieved when the steady state is always maintained at run-time. For a steady state, the temperature vectors are a direct consequence of constant power dissipation as follows:

$$[R(i,j)] * [Pj] = [Tj] \tag{2}$$

where R(i,j) is thermal resistance between cell i and j, Pj is power of cell j, and Ti is the temperature of cell i. Researcher have implemented two dynamic performance optimizations based on temperature and power. The power based optimization has a superior performance optimization by about 10%Temperature based optimization method can lead to wrong optimization result if the process with the high power consumption is allocated to a particular core. Some processes could not finish their jobs, but they still occupy cores and degrade overall processor speed (and throughput) since too much performance throttling is applied to those processes. Unlike the temperature based method, the proposed method by researchers based on estimated power consumption never violated the temperature constraint.

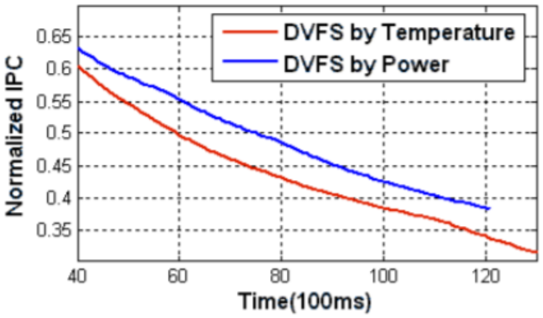


Fig. 2. IPC versus time for two different throughput optimization methods based on temperature and power [4]

## 2.2 Modelling energy consumption of servers in Data Centers

This paper focuses on modeling a server in an operational way. Therefore, the server is developed and validated using a communication server. Moreover, as virtualization is widely applied, the energy consumption will depend on the number of virtual machines that run on a server. Each virtual machine can be considered as a separate computer, with its operating system and allocated sources (CPU, memory, hard disk). The aforementioned sources are separated from the physical hardware (Figure 3). The management service of virtual machines is the hypervisor, which takes in charge VMs(virtual machines) start-up and shut down[5].

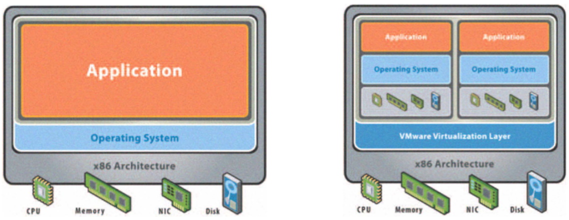


Fig. 3. Hardware and software architecture in a server of a data center before and after virtualization [5]

### 2.2.1 Methodology

There are two types of model for servers. The first considers the server as a set of equipments : CPU usage, hard disks, memory chips, motherboard, fans etc. The second type on the other hand, models the server as a single equipment, depending on parameters and inputs. To put it in other words, this is a black box modeling and the input of this model is the CPU usage.

Various models for estimating the power consumption of data centers have been proposed. However, the majority of these models present the percentage of CPU usage as a single unit. Whereas in this literature, the percentage of CPU is separated in two categories. The first one is the percentage due to servers self applications, while the second category is due to users applications. As a consequence, the energy consumed in a server will be modeled as multiple layers. Depending whether the server and the CPU is in idle or active mode. Three layers characterize this model, these layers are analyzed in the next section. Each layer will be added to the global power consumption of the server The power consumption of a server is presented as a linear function of the CPU usage. This CPU is considered as an input of the model (2)

$$Pserv = Pidle + (Pmax - Pidle)U \tag{3}$$

For the above equation, the Pidle stands for the power that is consumed when there is no load, *Pmax* is the server consumption in case of full load and *U* is the percentage of CPU usage due to the load.

### 2.2.2 Server power layers

The power consumption of the server is modeled as multiple layers. Depending whether the server is in idle or active mode. When the server is in idle mode, only servers applications run (management service, VM). Whereas, in the active mode, users application run on VMs. The first layer is the energy consumption due the physical hardware (hard disk, memory, fans). The power consumption of the aforementioned hardware components is considered fixed, since they dont present important fluctuations. In this state the server is in stand by mode(PstandBy), which means that the hardware components are supplied with power, but the server hasnt started yet.

The second layer is Pvirtualization which consists of *PhypervisorIdle* and *PvmIdle*. The CPU in this layer is in idle mode and it is called *Uoverhead*. Therefore, only a small percentage of the CPU is required. At first, the server starts but no virtual machines are launched. Consequently, hypervisor is in idle mode (*PhypervisorIdle*). In the next stage, the virtual machines are loaded by the hypervisor without running any application (*PvmIdle*). At this point, this overhead induces energy consumption without any productivity.

In the third layer, the consumption is caused by the applications that run on VMs (*Papplication*). In this state, the CPU usage (*Uactive*) as well as the power consumption (*Papplication*) depend on the requirements of the application. The Figure 4 shows the layers of the global power consumption.

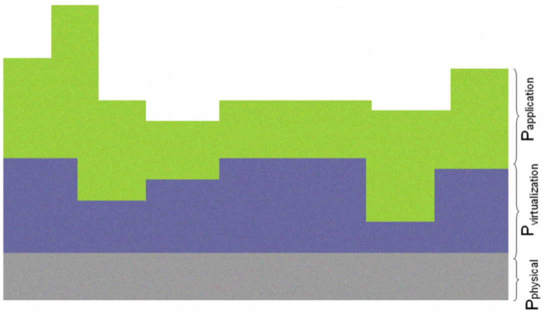


Fig. 4. Three major layers of server power consumption [5]

### 2.2.3 Experiments

Power measurements were taken when the server was in standby mode, when the hypervisor was launched, and finally when the VMs were loaded on the server without running any application. *PhypervisordIdle* is estimated 20W after a direct measurement on the server. PvmIdle is calculated by multiplying the number of virtual machines and the parameter A, which is estimated by measuring the power consumed after shutting down a VM.

The measurements that were carried out in idle mode proved that the relation between the *Uoverhead* and number of virtual machines is linear when the latter is below 64. Regarding the *Papplication*, a new parameter B should taken into account, in order to calculate the active percentage of the CPU. To estimate the value of B, is required to divide the percentage of CPU usage of 16,32 or 64 VMs in idle mode by the number of VMs. Finally the power consumption of a server can be calculated by the following equation(3),(4).

$$Pserv = PstandBy + Phypervisoridle + A * NVM \tag{4}$$

$$Pserv1 = Pserv + a * (Utotal - (Uactive * B)) \tag{5}$$

Where: *a'* is the relation between the power consumption, *NVM* stands for the number of virtual machines and *Uactive* is estimated by running Volano benchmark.

## 2.3 A Stochastic Model for Estimating the Power Consumption of a Processor

The stochastic model for estimating the power consumption of single-core and multi-core processors is a probabilistic approach, which estimates the power consumption of a workload of unknown size and type. For example, this model is most useful for internet servers where the size of the workload exhibits considerable fluctuations. While most models consider the relationship between the power consumption and the workload as deterministic, the stochastic model corresponds to reality and considers this relationship as imprecise. Hence, the workload and the power consumption are modeled as random variables.The aim of this approach is to determine the relationship between the aforementioned variables, by exploiting the monotonicity property of their distribution function.

This model provides a direct translation of the CPU activity level into power consumption level. In addition to the estimation of the instantaneous power consumption, it also estimates the statistics of the power consumed by a processor and relates it to the statistics of the workload. The purpose of the stochastic model is to estimate the power budget of a future workload, in order to decide in advance where and when to execute it. Furthermore, it aims to capture the magnitude and the frequency of power fluctuations.In addition, it can estimate the probability that a workload execution crosses a certain power threshold.

### 2.3.1 Theoritical Background

This probabilistic approach employs the statistics of CPU utilization. There is a strong correlation between the statistics of the processors workload and the statistics of the processors utilization. In addition, through repeated experiments is observed that there is also a strong correlation between the power consumption and the CPU utilization (Figure 5). Hence, examining the statistics of the processors utilization gives the possibility to estimate the power consumption of the processor due to the workload. The advantage of the stochastic model is that the statistical properties of one of the random variables can be sufficiently determined from the statistics of the other random variable. Since the stochastic model employs only the information of the statistic of CPU utilization, it makes the model lightweight and easily accessible in any platform. The estimation of the power consumption requires the actual measurement of the power. However, this model can bypass the measurement, once the relationship between the power consumption and the workload is established.

In order to establish the aforementioned relationship, a workload should be provided to the processor. The CPU will be modeled as a non-linear memoryless system with stochastic input. Then the study of the distribution or the density function of the CPU utilization and the power consumption of the processor during the execution period of the workload, will give the possibility to estimate the quantitative relationship between the CPU utilization and the power consumption.

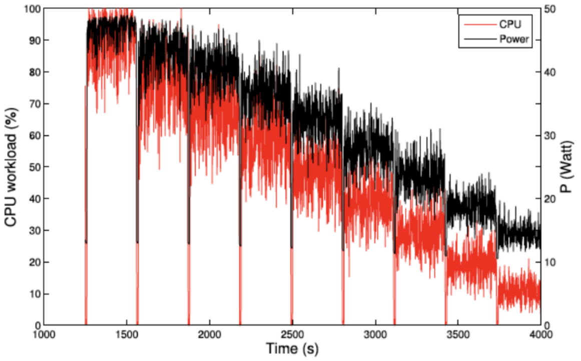


Fig. 5. The relationship between the CPU utilization and the power consumption of a D2581 Siemens-Fujitsu server running the SPEC Power (2008) benchmark [11].

### 2.3.2 Methodology

The methodology for estimating the relationship between the power consumption and the workload can be stated as follows: the distribution function of **w** is given and the aim is to find a function g(**w**) such that the distribution function of *p=g(w)* equals the function *F(p)*.

Where **w**, **p** are the random variables, *F* is the distribution function and *f* is the density function.Initially, a workload of known statistics is supplied to the processor. Therefore, the power consumption can be measured and analyzed. As a result, the relationship of the aforementioned can be established. Once that is achieved, the power consumption of any arbitrary workload can be efficiently estimated.

### 2.3.3 Single Processor

In order to establish the relationship between **w** and **p** of a single-core processor, a one hour uniformly distributed CPU-bound workload was supplied to the server. The first step is to generate the desired density function of the workload by picking time slots in which the CPU was fully utilized and measure the overall the overall power consumption of the server. The second step is to approximate the probability distribution function of the actual power consumption, using the nls curve fitting tool, in order to obtain *g(w)* using

$$p = Fp^{-}1(F(w)) \tag{6}$$

. In addition, five uniformly distributed workloads were generated in order to measure the actual distribution function of the power consumption and compare it to the estimated value (Figure 6).

The desired relationship between the CPU workload and the power consumption can be expressed as

$$\frac{p = -a2 + \sqrt{a^2 - 4a1 * (a3 - (w-b)/(c-b))}}{2 * a1} \tag{7}$$

Where: *b* and *c* are the lower and the upper bounds of the distributed workload. a1,a2,a3 are the experiment coefficients of the workload. Using this relationship is possible to determine the distribution and density of **p** in terms of the distribution and density of **w** for any workload.

### 2.3.4 Dual-core processor

In the multi-core processor a Multiple-Input-Single-Output (MISO) memoryless stochastic model is used to represent the relationship between the power consumption(**p**) and the workload(**w**). In this case the first step is to determine the overall workload of the processor, which is given as w = w1 + w2 +  + wn, where each w is the workload of each individual core. Compared to the single-core methodology, a uniformly distributed workload was generated for each core, *w1* and *w2* in order to generate the density function of the workload. The remaining steps are the same as in the single-core processor[11].

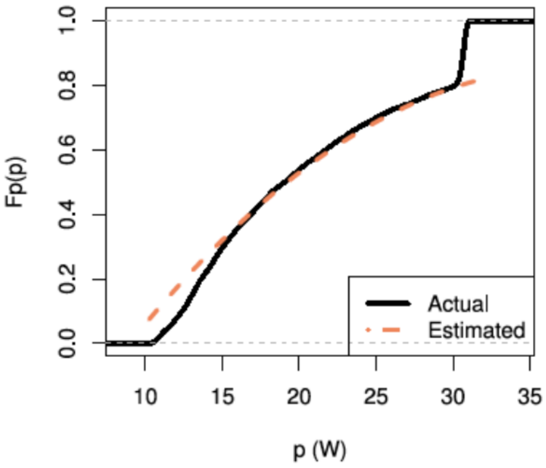


Fig. 6. The actual (measured) and estimated *F(p)* for the Intel E8500 [11]

## 2.4 System Power Estimation Using Processor Performance Events

It is considered by researchers that measurement of complete system power consumption can be done with the use of microprocessor performance counters. This has been based on the well-known fact that CPU power consumption is correlated to processor performance. However a new method in estimating power consumption could be the established through the use of the well-known performance-related events within a microprocessor such as cache misses and DMA transactions.

### 2.4.1 Complete system power model

Researchers considered extending the concept of using performance events as proxies for power measurement beyond the microprocessor to various computer subsystems. Even though microprocessors are typically the largest consumers of power, other subsystems constitute 40-60 percent of total power. The subsystems that are considered are : graphics processing unit (GPU), chipset, memory, I/O, and disk. If all of the subsystems are taken into consideration while providing means for power management, and not only the microprocessors, then it could have an significant impact on power and temperature. This approach is distinct since it uses events local to the processor, eliminating the need for sensors spread across various parts of the system and corresponding interfaces. Researchers showed that the microprocessor performance events can accurately estimate total system power[1]. This is very efficient because a model for estimating power consumption in this way would use no additional sensors or counters other than what the performance engineers have already incorporated[9].
Trickle-down power modeling provides an accurate representation of complete-system power consumption by relying on the broad visibility of system-level events to the processor.[2] The depiction in Fig. 7 represents a general server type system. The arrows flowing outward from the processor represent events that originate in the processor and trickle-down to other subsystems (L3 Miss, TLB Miss, MemBus Access, and Uncacheable Access). Arrows flowing inward such as Direct Memory Access (DMA) or bus master access and I/O interrupts may not be directly generated by the processor, but are nevertheless visible. [3]. An excellent iterative modeling procedure developed for estimating power from performance events utilizes linear and polynomial regression techniques to build power models for individual sub-

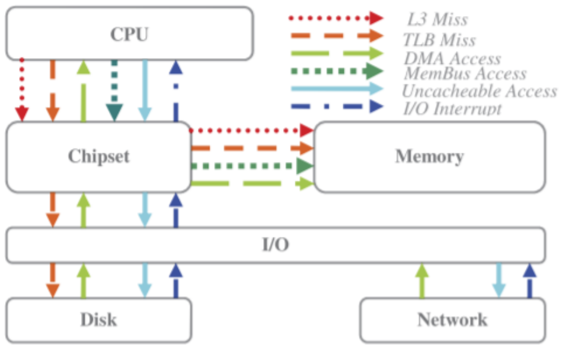


Fig. 7. Propagation of performance events [2]

systems. The user identifies workloads which target a particular subsystem (cache, system memory, disk) and performs regression modeling using performance events as inputs. The model is then applied to a larger set of workloads to confirm accuracy. Depending on the outcome, the process is repeated with alternate performance events as inputs. The modeling process involves several steps:

1. Measure subsystem-level power using subset of workloads [2].
2. Confirm that Coefficient of Variation is greater than a threshold alpha for the chosen workload. Tuning the model based on workloads with low variation of power may cause the process to include performance events that do not correlate well with power[2].
3. Based on domain knowledge, choose performance events, measurable by performance counters that are most relevant to the subsystem in question. The pool of candidate performance counters may need to be expanded if sufficient accuracy is not achieved[2].
4. Using the selected performance counter events as the input variables and subsystem power as the output variable, perform linear regression modeling. Multiple linear or polynomial regression may be used in subsequent iterations of algorithm if sufficient accuracy is not obtained using simple linear regression[2].
5. Calculate average error per sample. If less than the desired rho percent error cannot be achieved, a new performance event must be chosen. Select rho depending on the required model accuracy and time required for solution. Setting rho to a low (restrictive) value may extend the time taken to reach a solution. It may also prevent the process from finding a solution[2].
6. Assess the representativeness of the model by graphically comparing modeled versus measured power. This avoids the case in which statistical assessment cannot detect major errors such as those seen in Anscombes Quartet[2].
7. Using complete set of workloads calculate average error per sample. If less than the desired delta percent error cannot be achieved, choose a new performance event. Like rho , delta is selected according the accuracy and time-to-solution requirements[2].

## 3 Results

### 3.1 Results in Run-time Temperature-Based Power Estimation

The most efficient way considered by researchers to estimate power consumption in multi-core dies, is by using a software-based method to estimate each cores power consumption based on core temperature sensed with only one thermal sensor per core. Some researchers considered the idea of implementing a hot-spot temperature reconstruction method that recycles the computations performed for the core power estimation. In the end they optimize the throughput of thermally constrained multi-core processors by applying the estimated power [4]. This specific method it is said to lead to more optimal throughput with lower computational overhead than temperature-based optimization [4]. This estimation method of dynamic power consumption by using a thermal sensor exploits run-time power estimation values using each core's measured temperature from a single temperature sensor per core. According to the researchers experiment using 90nm technology, the proposed throughput optimization method provides 4% higher throughput than a temperature-based optimization method. Meanwhile, the reconstructed temperature based on the estimated power consumption results in less than 3% error in hot-spot temperature of a multi-core processor[4].

### 3.2 Data centers

Commercial servers as well as Simulink were used to validate the model (3) and (4). The voltage, current and power consumption of the server were measured 5 times per second with 5% precision. The estimation of power consumption of the aforementioned model was fairly accurate when the number of virtual machines was stable. The deviation between the estimated power and the actual measurement was 0.6% in the case of 32 VM and 0.3% in the case of 8 VM. In addition the model (2) is implemented in Simulink in which the load of the VMs can be dynamic(Figure 8) and the number of virtual machines can change (Figure 9). In both cases the power consumption was estimated fairly accurately [5].

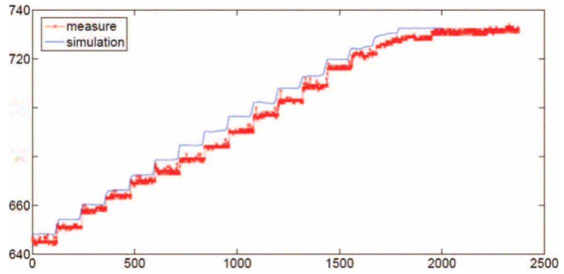


Fig. 8. Implementation of server model when IT load is changing [5]

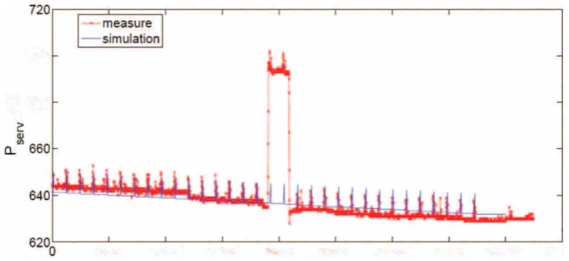


Fig. 9. Implementation of server model when the number of VM is changing [5]

### 3.3 Stochastic model

The validity of this model was tested by generating normally distributed and exponentially distributed workloads, SPEC power benchmark and Apache benchmark (Figure 9). The Apache benchmark operated in two extreme utilization regions, at 0% and 100%. Whereas the SPEC benchmark stressed the processor in different ways, with various size of workloads and power consumption fluctuations. The estimation error for the Apache benchmark was approximately 2.7% for both single-core and multi-core processors, while for the SPEC benchmark the error increased to 7.37% for the single-core and 5.2% for the multi-core processor [11].

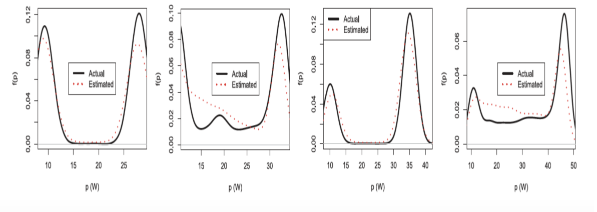


Fig. 10. The actual and estimated *f(p)* of the power consumption of the Intel E8500 processor when executing, from left to right, the Apache benchmark (single-core mode), the SPEC power benchmark (single-core mode) the Apache benchmark (dual-core mode), and the SPEC power benchmark (dual-core mode)[11].

### 3.4 Results for System Power Estimation Using Processor Performance Events

Predicting complete system power consumption using processor performance events has a high degree of feasibility. All these events had a trickle down effect on them, and taking advantage of this situation is the key success of this particular model of estimating power consumption. Power consumption in subsystems including memory, chipset, I/O, disk, and microprocessor are correlated to these events. Subsystems farther away from the microprocessor require events more directly related to the subsystem. DMA events related to memory models take into account the activity that does not originate in the microprocessor. It is shown that complete system power can be estimated with an average error of less than 9% for each subsystem using performance events that trickle down from the processing unit [3].

## 4 Comparing models for estimating the power consumption of systems

The four models that are presented and analyzed in this study aim to estimate the power consumption of systems. While all of the aforementioned models have the same goal, the approach of each model differs significantly. Apparently, the reason that there are differences between the methods is due to the fact that each model is used for different purposes. The stochastic model examines the proportionality between the power that is consumed by a processor and the work it accomplishes, in order to estimate the power budget of a future workload and the probability of crossing a certain threshold. Whereas in data centers, the energy that is consumed is modeled as accumulated layers, depending on the CPU usage. The purpose of this model is to compute the power that is required to run a an application. The processor performance events model analyzes the microprocessors performance counters for on-line measurement of complete system power consumption [2]. The Runtime Temperature-Based model uses software-based techniques to estimate the power consumption of multi-core processors based on each cores temperature in order to improve the processor's efficiency.[8].

Consequently, the choice of a model partially depends on the needs of each case. However, there are criteria that can be followed to decide which model best meets the requirements of each case. For instance, the complexity of a method should be taken into consideration. Even though monitoring the events emitted by hardware components provide useful insight into the activities of the various hardware components, the architecture changes from CPU to CPU, which increases the complex of this model. The method that uses thermal sensors to estimate the power consumption is the least complex. However the detection of the temperature is not very efficient since the temperature changes may occur significantly later than the events that cause it. Another criteria is to choose the model with the lowest error rate when estimate power consumption. We noticed that the method which uses the temperature of each core has less than 3% estimation error. A similar rate is obtained when utilizing the Apache benchmark in the Stochastic model where the error was approximately 2.7% for both single-core and multi-core processors. Whereas in data centers the deviation between the estimated power and the actual measurement was 0.6% in the case of 32 VM and 0.3% in the case of 8 VM. There should be taken into consideration also the importance of cost minimization. For this reason we have to attribute credit to the method which eliminated the need of power sensing hardware, such as sensors, by using existing on-chip performance event counters for estimating system power consumption. This solution is far more desirable than the one in which researchers presented an estimation method of dynamic power consumption using a thermal sensor.

In the end there can be noticed the fact that the different approaches lead to different results even in the pursuit of the same goal. By balancing the advantages and disadvantages presented above we could state that the most efficient method is the one that employs the statistics of CPU utilization in order to establish the relationship between the workload and the corresponding power consumption of a processor. Together with the method that involves using processor performance events, it can lead to a complete system power estimation.

## 5 Conclusion

In conclusion we can state that all methods analyzed in this research paper provide a good understanding of how estimation of power consumption in a system should be done. It can either use software-based techniques to estimate the power consumption of multi-core processors based on each cores temperature. Another possibility is by employing the statistics of CPU utilization in order to establish the relationship between the workload and the corresponding power consumption of a processor. It can also analyze microprocessor performance counters for on-line measurement of complete system power consumption. And lastly it can model the servers in data centers. The most efficient models, as presented in the previous section, provide the lowest estimation error, the highest focus on different components and are the most complex. We considered that there should also be considered the combining the two models for a more effective result.

## Acknowledgements

The authors wish to thank Rein Smedinga, Michael Biehl and Femke Kramer as the editors of the SC@RUG 2018 proceedings.

## References

[1] F. Bellosa. The benefits of event-driven energy accounting in power-sensitive systems. Proc. Ninth Workshop ACM SIGOPS European Workshop: Beyond the PC: New Challenges for the Operating System, pp. 37-42, September 2000.

[2] W. Bircher and L. John. Complete system power estimation: A trickle-down approach based on performance events. IEEE Intl Symp. Performance Analysis of Systems and Software, pp. 158-168, April 2007.

[3] W. L. Bircher and I. Lizy K. John, Fellow. Complete system power estimation using processor performance events. IEEE TRANSACTIONS ON COMPUTERS, VOL. 61, NO. 4, April 2012.

[4] C. C. P. C. A. D. Y. H. H. Dongkeun Oh, Nam Sung Kim. Runtime temperature-based power estimation for optimizing throughput of thermal-constrained multi-core processors. National Taiwan University.

[5] V. D. S. B. Ghaith Warkozek, Elisabeth Drayer. A new approach to model energy consumption of servers in data centers. leIT 2012, 2012.

[6] Z. Gud. Aphs: Thermal aware unified physical level and high level synthesis. Proc. IEEE ASP-DAC, pp. 879-885 2006.

[7] C. Isci and M. Martonosi. Runtime power monitoring in highend processors: Methodology and empirical data. Electrical Eng. Dept., Princeton Univ., Princeton, NJ, USA, Tech. Rep., December 2003.

[8] S. Murali. Temperature control of high performance multi core platforms using convex optimization. Proc. IEEE DATE, pp.110-115 2008.

[9] D. I. P. Ranganathan, P. Leech and J. Chase. Ensemble-level power management for dense blade servers. Proc. 33rd Ann. Intl Symp. Computer Architecture (ISCA 06), pp. 66-77, June 2006.

[10] R. Teodorescu. Variation-aware application scheduling and power management for chip multiprocessors. Proc. IEEE/ACM ISCA, pp.363-37 2008.

[11] I. Waltenegus Dargie, Senior Member. A stochastic model for estimating the power consumption of a processor. IEEE TRANSACTIONS ON COMPUTERS, VOL. 64, NO. 5, May 2015.

# Discriminative vs. Generative Prototype-based classification

Jan Boonstra, Nadia Hartsuiker

**Abstract**—Learning Vector Quantization (LVQ) algorithms are powerful and widely popular models for class discrimination of vectorial data. However, classification of vectorial data remains a challenging topic. GLVQ has been proposed as a generalised approach to the LVQ algorithms. Since then various extensions to GLVQ have been introduced as ways to improve GLVQs ability to more accurately discriminate between classes, as well as to better represent the underlying data. [4] [3]

In this paper we compare several proposed variants of GLVQ in their ability to fulfil several requirements for optimal use of GLVQ algorithms, including discriminative power, class-representativity and border-sensitivity. Several data sets are used on each of the algorithms to reliably test these properties.

**Index Terms**—Intelligent systems, machine learning, classification, LVQ, GLVQ, BS-GLVQ.

## 1 Introduction

Learning Vector Quantization (LVQ) algorithms are powerful and widely popular models for class discrimination of vectorial data. However, classification of vectorial data remains a challenging topic. Sato and Yamada [7] have introduced Generalised Learning Vector Quantization (GLVQ), where reference vectors are updated based on the steepest decent method, minimising the cost function. Hammer et al. [3] compare adjusted versions of GLVQ, where the cost function is extended to make resulting reference vectors more representative of their class. A border sensitive approach (BS-GLVQ) where data points close to the class border contribute more significantly to prototype adaption is introduced by Kaden et al. [4]

With both class-representative GLVQ and BS-GLVQ an attempt has been made to solve problems GLVQ has with its interpretation and stability. While the mentioned papers do provide results regarding the performance of the algorithms they introduce, comparing the actual performance is difficult since different data sets are used to demonstrate each approach. In this research we will compare the cost functions of GLVQ, class-representative GLVQ and BS-GLVQ on their classification performance and their positioning and interpretation of prototypes.

In this paper we will first give an overview of the work that was done in the the field of Vector Quantization by explaining the algorithms mentioned above. After that we will present the results we got after implementing these algorithms and testing them on several data sets.

In order to generate comparable results, each algorithm will be tested using the same data sets. This will allow for a comparison of the performance of class-representative GLVQ and BS-GLVQ. Results of the performance of GLVQ will be used as a reference. In addition, comparing results of GLVQ with class-representative GLVQ and BS-GLVQ replicates the research of Hammer et al. [3] and Kaden et al. [4].

## 2 Background

This section will give an overview of the different classification algorithms that were studied. The distance function used in each of the following subsections is the squared Euclidean distance.

### 2.1 LVQ

Learning Vector Quantization (LVQ) was proposed in 1995 by Kohonen et al. [5] This type of LVQ became later known as LVQ1, after improved versions such as LVQ2.1 were proposed. The idea of LVQ and all algorithms based on it is to generate a set of so-called reference-vectors based on the provided data set that can discriminate between the different classes of the data. LVQ aims to optimise the locations of these vectors in such a way that each point in the data-set is of the same class as the reference vector it's closest to. This way the class of a new data point can be predicted by finding the nearest reference vector. We will give first a brief overview of the functionality of LVQ1.

For each data vector $x_i$ a reference vector $w$ is selected, such that $w$ is the nearest reference vector from $x_i$. Depending on whether $x_i$ was classified correctly before, $w$ is either attracted to and repulsed from $x_i$, according to the following update rules.

$$w = w + \alpha(x_i - w) \tag{1}$$

if x is classified correctly.

$$w = w - \alpha(x_i - w) \tag{2}$$

if x is classified incorrectly.

Here $\alpha \in [0,1]$ is a scaling factor that decreases as a function of time. Later Kohonen et al. also introduced

- *Jan Boonstra is a student Computing Science at the University of Groningen, E-mail: j.boonstra.5@student.rug.nl.*
- *Nadia Hartsuiker is a Computing Science student at the University of Groningen, E-mail: N.L.Hartsuiker@student.rug.nl.*

an improved version of LVQ1 called LVQ2.1. In LVQ2.1 2 reference vectors are updated each iteration. For each data vector $x_i$ two reference vectors $w_1$ and $w_2$ are selected, such that $w_1$ is the nearest reference vector of the same class as $x_i$ and $w_2$ is the nearest reference vector of a different class than $x_i$. $w_1$ and $w_2$ are attracted to and repulsed from $x_i$ respectively according to these rules:

$$w_1 = w_1 + \alpha(x_i - w_1) \tag{3}$$

$$w_2 = w_2 - \alpha(x_i - w_2) \tag{4}$$

These update rules are only applied when the distances $d_1$ and $d_2$ from $x_i$ to the reference vectors $w_1$ and $w_2$ respectively are within a certain window defined by the following condition:

$$min(\frac{d_1}{d_2}, \frac{d_2}{d_1}) > s \tag{5}$$

However, neither LVQ1 nor LVQ2.1 has a system in place in to make sure the reference vectors converge towards an optimal configuration. This means that depending on the initial configuration and the data set the reference vectors may diverge, meaning that the vectors keep moving around faster and faster, which means they will never end in an optimal configuration.

### 2.2 GLVQ

In 1996 a generalised version of LVQ was proposed [7]. This algorithm, called GLVQ, was introduced to solve the problem with diverging vectors as described above. Sato et al. introduce the following cost function for GLVQ:

$$E = \sum_{i=1}^{N} f(\mu(x_i)) \tag{6}$$

where $f(\mu)$ is a monotonically increasing function and N is the number of data points in the training set and the relative distance function $\mu(x)$ is defined as:

$$\mu(x) = \frac{d_1 - d_2}{d_1 + d_2} \tag{7}$$

If this relative distance function is negative, then $x_i$ is classified correctly, since the nearest reference vector of the same class is closer to $x_i$ than the nearest vector of any other class. If $\mu(x_i)$ is positive however, $x_i$ is classified incorrectly, so $w_1$ and $w_2$ are updated with attractive and repulsive forces with GLVQ's update rules:

$$w_1 \leftarrow w_1 + \alpha \frac{\delta f}{\delta \mu} \frac{d_2}{(d_1 + d_2)^2}(x_i - w_1) \tag{8}$$

$$w_2 \leftarrow w_2 - \alpha \frac{\delta f}{\delta \mu} \frac{d_1}{(d_1 + d_2)^2}(x_i - w_2) \tag{9}$$

For $f$ Sato et al. propose the identity function as well as a sigmoid function. This research will focus on the sigmoid variant:

$$f(\mu) = \frac{1}{1 + e^{-\mu}} \tag{10}$$

The corresponding derivative of $f$, which is needed for the update rules then becomes:

$$\frac{\delta f}{\delta \mu} = f(\mu) \cdot (1 - f(\mu)) \tag{11}$$

In their paper, Sato et al. prove that their GLVQ satisfies the convergence condition, while LVQ2.1 does not. For LVQ1 convergence cannot be shown explicitly, since it does not have a cost function.

### 2.3 Class-representative GLVQ

One feature of LVQ and its variants is that the resulting reference vectors are often in locations that are representative of the class that they belong to, such that these vectors are easily interpretable. This is not always the case, however, and when new variants of LVQ are introduced, class-representativity is usually not taken into account in the cost function. For this reason an extension to GLVQ was proposed by Hammer at al. [3] that tries to make reference vectors more representative. This new cost function $E$ consists of two parts: $E_{discr}$ and $E_{repr}$. Here $E_discr$ is the normal cost function of GLVQ and $E_{repr}$ is a cost function that is based on how representative the reference vectors are. Their extended cost function is the following:

$$E = (1 - \alpha) \cdot E_{discr} + \alpha \cdot E_{repr} \tag{12}$$

where $\alpha$ is the weight factor that decides the ratio between the normal cost function $E_{discr}$ and the class-representativity function $E_{repr}$. Furthermore, Hammer et al. introduce the following representativity cost function:

$$E_{repr} = \sum_j d^+(x_j) \tag{13}$$

Where $d^+(x_j)$ denotes the distance from $x_j$ to the nearest reference vector of the same class. From this, the following update rule arises:

$$w_1 \leftarrow w_1 + \alpha(x_i - w_1) \tag{14}$$

### 2.4 Border-sensitive GLVQ

Kaden et al. compared GLVQ to an alternative type of machine learning, Support Vector Machines (SVMs), and noted that their differences in behaviour made SVMs more desirable than GLVQ for many applications. In particular, they focus on SVMs' ability to detect class borders. In their paper they propose an extension to GLVQ that makes it more sensitive to class borders. In other words, data points near the border of their respective class influence the locations of the reference vectors more heavily than data points far away from those borders. They detail two approaches to implementing this extension.

The first approach adds a penalty function to the cost function of GLVQ that forces the reference vectors to move closer to the class borders. The second approach aims to make GLVQ more border-sensitive through changing the $f(\mu)$ function to include a sensitivity factor. In our research we will use the latter approach.

For this approach, Kaden et al. propose the following function $f_\theta(\mu)$ to replace $f(\mu)$ as it was originally introduced for GLVQ:

$$f_\theta(\mu) = \frac{1}{1 + e^{-\frac{\mu}{2\theta^2}}} \tag{15}$$

Where $\theta \in (0, 1]$ is a preset value determining the border sensitivity, with a lower value of $\theta$ making the algorithm more border-sensitive. The corresponding derivative of $f$, which is needed for the update rules then becomes:

$$\frac{\delta f_\theta}{\delta \mu} = \frac{f_\theta(\mu) \cdot (1 - f_\theta(\mu))}{2\theta^2} \tag{16}$$

## 3 Experiments

GLVQ, Class-representative GLVQ and Border-senitive GLVQ have been implemented and will each be used on the same data sets to generate comparable results.
To systematically test the performance of the algorithms described in the previous section, three data sets have been chosen from the UCI Machine Learning Repository [2]: a data set on Iris flowers, a data set on contraception use of women and a data set on the occurrence of breast cancer.

### 3.1 Performance using data on Iris flowers

The first data set used is on Iris flowers and consists of 150 instances evenly distributed over three classes. Each element has four features representing the length and width of the sepal- and petal of the flower. The class is defined by the subspecies the flower belongs to.

As a preprocessing step the data has been normalised by computing the standardised z-score for each element, meaning that the mean of each feature is centred to 0 and the feature values have been scaled such that the standard deviation is 1.

A fifth of the data is selected by taking the first twenty percent of elements of each class and used to train the classifiers. For each class three prototypes are created. The remaining eighty percent of the data is uses to evaluate performance.

The performance of each approach is measured in terms of accuracy of classification and in terms of representation value. Accuracy of classification can be compared by computing the classification error. To quantify representation value, Hammer et al. [3] have come up with a ratio $R$, defined as follows:

$$R = \frac{1}{C}\sum_{c_k}\sum_{j:c(x_j)=c_k}\frac{d^+(x_j)}{\sum_{i:c(x_i)=c_k} d(x_i,\mu_{c_k})} \tag{17}$$

Where $C$ is the number of classes in the data set, $x_j$ are data points belonging to the training set, $x_i$ are data points belonging to the full set and $\mu$ is the class mean.

#### 3.1.1 Class representative GLVQ

To test the performance of class representative GLVQ, the value $\alpha$ discussed in previous sections is adjusted. GLVQ is executed muliple times with different values of $\alpha$ between one and zero. Zero corresponds to 'regular' GLVQ, and as $\alpha$ increases the 'Class representative' aspect of the algorithm increases. The result for 300 iterations can be found in Figure 1.

From the Figure can be observed that the increase in $\alpha$ does not lead to a better representativity value $R$. In addition, the classification error increases as $\alpha$ increases, meaning a lower classification accuracy.

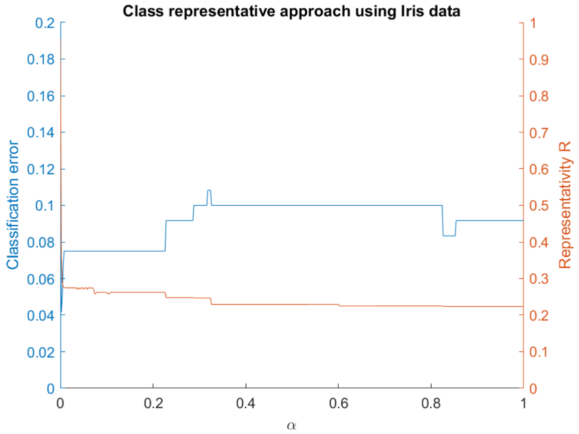


Fig. 1: Performance of class representative GLVQ on iris data.

In figure 2 the earlier defined cost function is displayed by plotting the value of the cost function of the last iteration for different values of $\alpha$. It appears that the final value of the cost function decreases linearly as $\alpha$ increases. In this case a higher degree of 'class representativity' results in reference vectors that better fit the training data. This is, however, not reflected in the results when the performance is tested with 'new' data. (see figure 1).

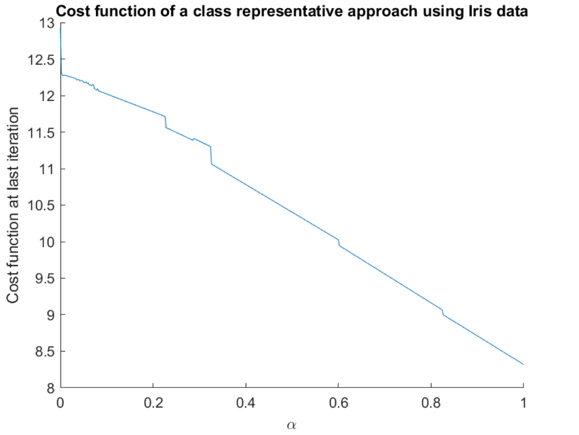


Fig. 2: Cost function of class representative GLVQ on iris data.

#### 3.1.2 Border sensitive GLVQ

The performance of border sensitive GLVQ is determined by adjusting the previously discussed $\theta$ in the domain [0,1] where a value of one corresponds to 'regular' GLVQ and the 'border sensitivity' increases as $\theta$ decreases. The result for 300 iterations can be found in Figure 3. Here can be observed that there is no major change in the representativity value $R$ as $\theta$ is altered. The classification accuracy appears to be optimal for values of $\theta$ between approximately 0.20 and 0.34. Note that the result at $theta = 1$ corresponds to the result at $alpha = 0$ in figure 1, as both are the result of 'regular' GLVQ.

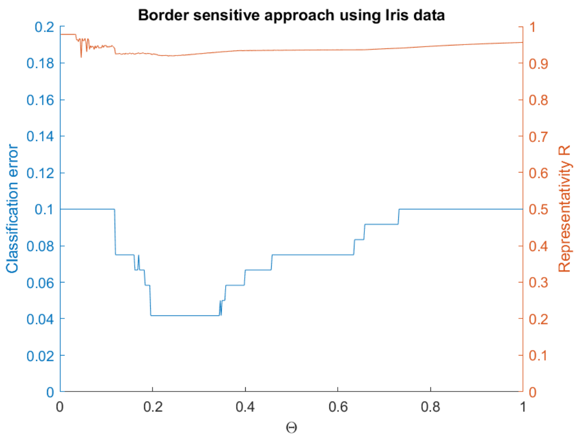


Fig. 3: Performance of BS-GLVQ on iris data.

In figure 4 the cost function has been plotted in a similar fashion as in the previous section: The value of the cost function in the last iteration has been plotted for different values of $\theta$. If we omit the values of $\theta$ at the border of the interval, it appears that the cost function converges to its minimum as the value of $\theta$ decreases. When the degree of border sensitivity of the algorithm increases, the feature vectors appear to represent the training data better. This is reflected when using new data in the classification error in figure 3, though not very obviously.

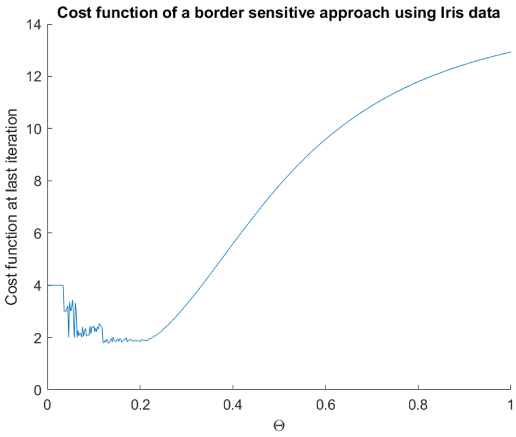


Fig. 4: Cost function of border sensitive GLVQ on iris data.

## 3.2 Performance using data on contraceptive methods

The second data set used is on contraceptive methods chosen by women. It consists of 1473 instances, each with nine features describing circumstances in women's lives. Defined classes are the use of either no birth control, short-term birth control or long-term birth control.

Preprocessing has been done in the same way as for the iris data. The data has been normalised and the first fifth of data points of each class is selected to train the classifiers. For each class three prototypes are initialised.

### 3.2.1 Class representative GLVQ

The Classification error and the class representativity $R$ for different values of $\theta$ are displayed in Figure 5. Here we can see that changes in $\theta$ have very little impact on both the classification error and the representativity $R$.

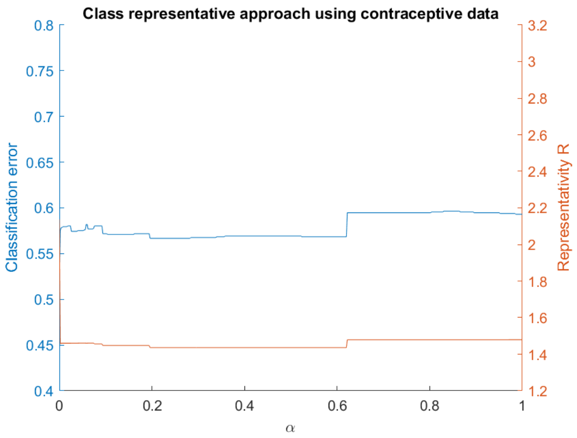


Fig. 5: Performance of class representative GLVQ on contraceptive data

The cost function has been plotted in figure 5 in the same way as for iris data. There appears to be a linear increase of the cost function as $\alpha$ and thus the class representativity increases. This is not reflected in the results when using new data in figure 5.

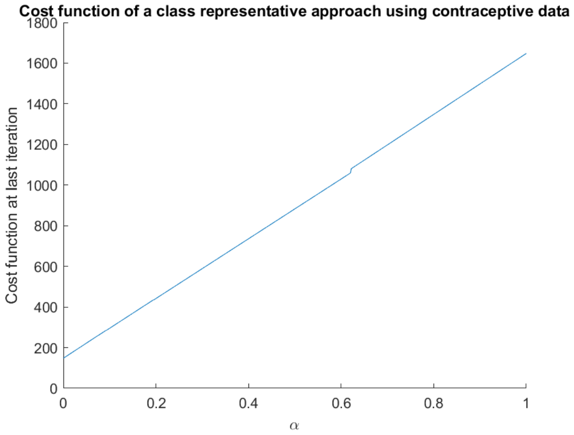


Fig. 6: Cost function of class representative GLVQ on contraceptive data

### 3.2.2 Border sensitive GLVQ

Results of the performance of border sensitive GLVQ while using the contraceptive data are plotted in Figure 7. The classification accuracy appears to improve slightly when the border sensitivity is increased by lowering $\theta$. However, after $\theta$ has been lowered to approximately 0.6, increasing the border sensitivity even more, lowering $\theta$ further has a strong negative impact on the classification error.

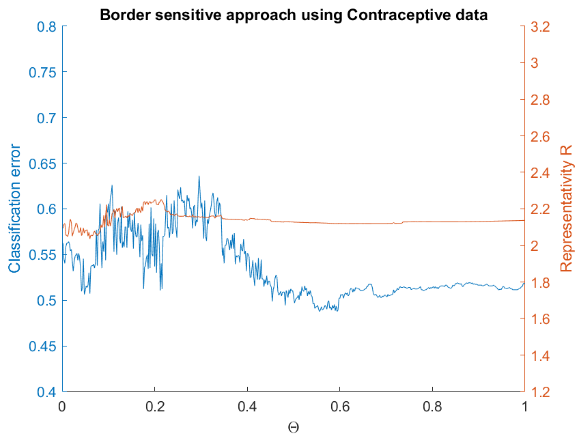


Fig. 7: Performance of border sensitive GLVQ on contraceptive data.

The cost function plotted in figure 7 appears to remain constant exept for low values of $\theta$, and a high degree of border sensitivity, where the result fluctuates.

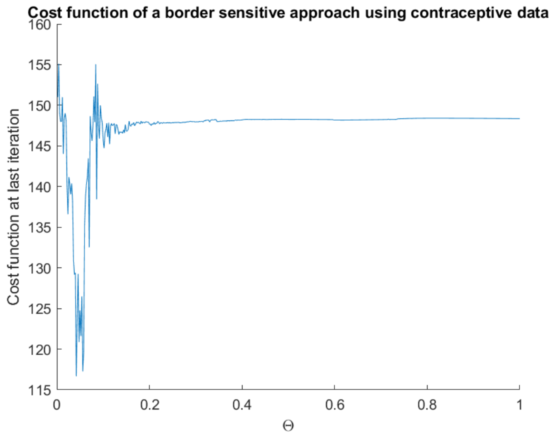


Fig. 8: Cost function of border sensitive GLVQ on contraceptive data.

Note that for both the class-representative and the border sensitive approach the classification error is very high. In many cases not even half of the data points is classified correctly. This classification error differs greatly from practical applications where much higher classification accuracy is used. One could argue that this classification error differs so much, and the result for this particular data set is so poor, that no meaningful conclusions can be drawn.

## 3.3 Performance using breast cancer data

The third and final data set used is a breast cancer database obtained from the University of Wisconsin Hospitals, Madison from Dr. William H. Wolberg, also used in [1], [6] and [8]. The database provides an ID-codenumber, eight attributes and the class being the diagnosis of either a benign or a malignant tumour. The database contains 699 elements, of which 16 have a missing feature. These have been excluded from further analysis, leaving 683 elements.

### 3.3.1 Class representative GLVQ

Results of the performance of class-representative GLVQ while using the breast cancer data are plotted in Figure 10. From these plotted results it becomes clear that the classification error fluctuates heavily based on the value of $\alpha$. Around $\alpha = 0.2$ both the classification and the representativity have a minimum value.

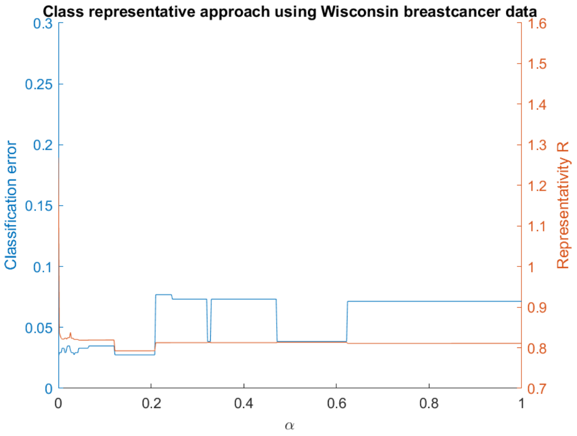


Fig. 9: Performance of class representative GLVQ on breast cancer data.

The cost function displayed in figure 10 shows, similar to the cost function of contraception data, a linear increase as $\alpha$, the class representativity, increases. This is not reflected in the results in figure 10 where new data is used.

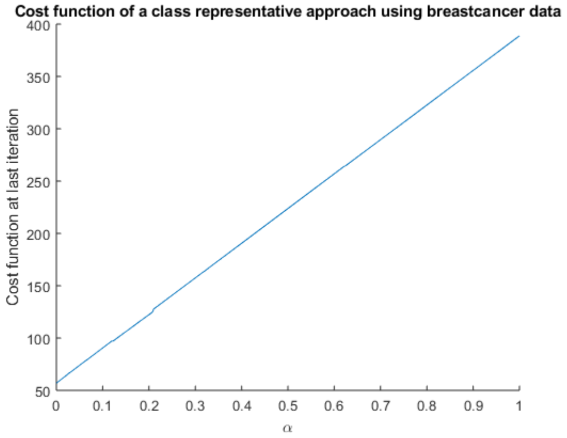


Fig. 10: Cost function of class representative GLVQ on breast cancer data

### 3.3.2 Border sensitive GLVQ

In figure 11 the results of Border Sensitive GLVQ when using the breast cancer data are plotted.

Both the classification error and the representativity $R$ are very stable, regardless of the value $\theta$ used.

The classification error here is very stable, regardless of the value $\theta$ used. There do occur slight changes in the representativity. $R$ has a minimum value around $\theta = 0.25$, but rises when $\theta$ approaches 0 or 1.

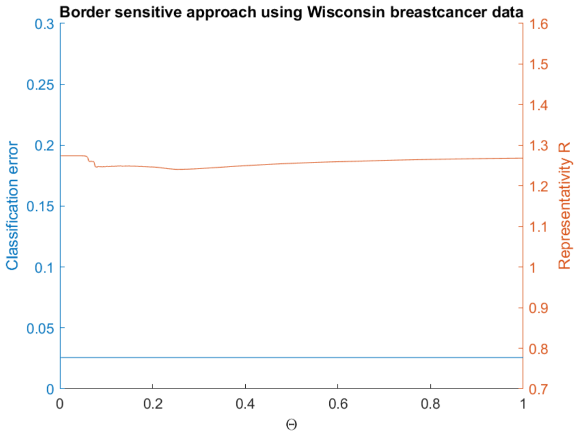


Fig. 11: Performance of BS-GLVQ on breast cancer data

The cost function in figure 12 shows similar behaviour as for the iris data. The cost function appears to converge to a minimum as $\theta$ decreases and the border sensitivity increases. This is not reflected in the results when using new data. (figure 12)

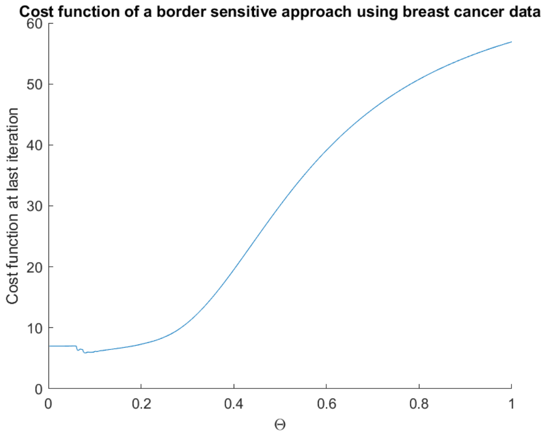


Fig. 12: Cost function of BS-GLVQ on breast cancer data.

## 4 Conclusion and Discussion

We have systematically tested the performance of varying degrees of class representative- and border sensitive GLVQ. Three datasets from the UCI Machine Learning Repository [2] have been used for this purpose: a data set on Iris flowers, a data set on contraception use of women and a data set on the occurrence of breast cancer.

When it comes to class representative GLVQ, there does not appear to be a pattern in the results from the used data. The class representativity $R$ remains fairly constant for a varying $\alpha$, and the slight changes that occur in $R$ are at different values of $\alpha$ in each data set. The same holds for the classification accuracy expressed in the classification error. Though there are some almost step wise fluctuations, these appear to be edge cases where a few elements are on the boundary of being classified either rightly or wrongly and do not show a clear pattern. The cost function appears to be linear for all (training) data sets. However, for the iris data, an increasing $\alpha$ corresponds to a lower value for the cost functions, while for the other two (training) data sets, a higher $\alpha$ corresponds to a higher value for the cost function.

For border sensitive GLVQ there also do not appear to be any patterns when it comes to class representativity and classification accuracy. When running the algorithm for the iris data, high values for classification accuracy (or low values for classification error) occur between a $\theta$ of approximately 0.20 and 0.34, while these values of $\theta$ give very high classification errors using the contraceptive data. The graph of the classification error of breast cancer data is completely flat. Values for the class representativity hardly change when $\theta$ changes. It does appear to be the case that slightly higher values of $R$ occur for low values of $\theta$, however, this is very minimal and is most likely disproportionate to changes in the classification error. The cost function does seem to have a set pattern, it appears to converge to a minimum as $\theta$ decreases. This is similar to what Kaden et al. [4] have found in their paper.

While based on the data found in this paper, no conclusions can be drawn on the performance of border sensitive GLVQ and class representative GLVQ, it does show that the same algorithm can generate very different results in terms of performance when applied to different data sets. This is something that should be taken into account when doing further research, as many researchers often limit themselves to a very small number of data sets to test their methods.

## Acknowledgements

We would like to thank prof. dr. Biehl for the comments on the draft version, the provided suggestion for the content of this paper and the valuable recommendation of the UCI machine learning repository where we retrieved the used data sets.

## References

[1] K. P. Bennett and O. L. Mangasarian. Robust linear programming discrimination of two linearly inseparable sets. *Optimization methods and software*, 1(1):23–34, 1992.

[2] D. Dheeru and E. Karra Taniskidou. UCI machine learning repository, 2017.

[3] B. Hammer, D. Nebel, M. Riedel, and T. Villmann. Generative versus discriminative prototype based classification. In T. Villmann, F.-M. Schleif, M. Kaden, and M. Lange, editors, *Advances in Self-Organizing Maps and Learning Vector Quantization*, pages 123–132, Cham, 2014. Springer International Publishing.

[4] M. Kaden, M. Riedel, W. Hermann, and T. Villmann. Border-sensitive learning in generalized learning vector quantization: an alternative to support vector machines. *Soft Computing*, 19(9):2423–2434, 2015.

[5] T. Kohonen. Learning vector quantization. In *Self-Organizing Maps*, pages 175–189. Springer, 1995.

[6] O. L. Mangasarian, R. Setiono, and W. Wolberg. Pattern recognition via linear programming: Theory and application to medical diagnosis. *Large-scale numerical optimization*, pages 22–31, 1990.

[7] A. Sato and K. Yamada. Generalized learning vector quantization. In *Advances in neural information processing systems*, pages 423–429, 1996.

[8] W. H. Wolberg and O. L. Mangasarian. Multisurface method of pattern separation for medical diagnosis applied to breast cytology. *Proceedings of the national academy of sciences*, 87(23):9193–9196, 1990.

# Face Clustering: A Comparative Study

Steven Farrugia S3362930, Amey Bhole S3411427

**Abstract**—A common problem encountered by researchers is constructing clusters to precisely identify the respective individual identities present in a large dataset containing unlabeled face images, videos, noise, camera motion, illumination and viewpoint variations. The problem may be characterized by two key components; face representation and determining similarity in order to cluster faces. A number of situations ranging from social media to criminal investigations [12] have further stimulated the evolution of numerous face clustering techniques. Y. Shi et al. proposed a face clustering method, using ResNet for feature representation and constructed a clustering algorithm called Conditional Pairwise clustering (ConPaC) [21] which outperformed well-known clustering algorithms. V. Sharma et al. highlights the importance of feature representation in clustering face tracks of an identity within a video [20]. Different aggregate methods were employed on VGG Deep face (VDF) model after which the performance on K-means and bottom-up hierarchical agglomerative clustering were evaluated. Seeking to identify the best internal and external cluster quality, C. Otto et al. developed an efficient and powerful Rank-Order clustering algorithm which was applied to both subsets of and the full data-set containing 123 million face images [16].

In this paper, we highlight and contrast a number of effective and adapted techniques implemented in the above researches on large-scale face data-sets, the challenges encountered, and the variety of metrics used to quantify the quality of the clusters produced. We also suggest possible improvements on top of already existing implementations to improve both computation time and clustering performance.

**Index Terms**—Face Representation, Clustering techniques, Convolutional Neural Networks (CNN), Clustering validation

## 1 Introduction

Clustering face images according to unique identities may be characterized by two key applications; i) grouping a large set of face images without any or commonly incorrect associated external labels to define the image content, and ii) indexing in order to efficiently retrieve faces on a large scale. In other words, we may define the problem as face representation coupled with the measures by which similar faces may be grouped together. Often, the number of clusters or unknown identifies may vary between hundreds to hundreds of millions which formulates both a computation and cluster quality optimization problem.

If we take social media into consideration, an average of 350 million photos are reportedly uploaded on Facebook per day [1], from which we may assume a large quantity are of people. Although one may provide relevant tags, these are generally incomplete or inaccurate. To organize this large data-set, one may consider grouping the faces present within these images into distinct identities. This problem has already been encountered during forensic investigations such as the Boston Marathon Bombing [12]. During the time-sensitive investigation, tens of thousands of images as well as videos needed to be analyzed. Other relevant examples include identifying criminals, perpetrators and victims to child exploitation.

In the aforementioned situations, one would expect the number of images per identity to be unbalanced (some people may have more photos than others), which is an issue for certain clustering algorithms such as k-means. Manually ascertaining the cluster quality on a large scale can also be too time consuming. Therefore a number of cluster quality measures, taking into account both internal and external cluster measures, help ensure the effectiveness of the applied face clustering algorithm.

This review paper has been composed in such a way that we first explain the approach defined or proposed by the three papers we reviewed in sections 2.1, 2.3, 2.2 separately. This is then followed by the respective results obtained per paper (sections 3.1,3.2,3.3) on which we then highlight and identify similarities and differences between the methods executed (in section 4) based on a number of factors and measures; such as scalability, F-measure, runtime and computational complexity.

## 2 Methods

In this section we provide an overview of the clustering approaches used throughout 3 research papers; face representation, clustering algorithms, cluster evaluation and training/evaluation data-sets. A more in-depth interpretation was provided for the clustering algorithm the researchers used/adapted.

### 2.1 A simple and Effective for Face Clustering in TV Series

#### 2.1.1 Face Representation

V. Sharma exploited the feature representation of the VGG Deep Face (VGF) model [13] pre-trained on 2.6 Million face images. Applying it to the three face clustering data-sets, specified in table 1, a track-level representation was produced, combining all the respective frames together in a track; characterized by a single feature map extracted from the last fully-connection layer (fc) for each image in the track. Face tracks were gathered via a tracking-by-detection method using a particle filter [5]. During the feature extraction procedure, applied on multiple videos, the respective RGB images were re-sized to 227 x 277 following which they were mean-subtracted by a value of 128. Specifically, the fc7 features of the network were extracted, producing descriptor vectors in 4096 dimensions. A broad overview of the full architecture may be seen in figure 1.

#### 2.1.2 Aggregate functions & Clustering

Three different aggregation functions $\phi$ were investigated; i) element-wise average of track, ii) element-wise maximum of track and iii) element-wise multiplication of track.

A comparison of the three aggregation functions was applied on the feature maps ( extracted from the last fully-connected layer) defined as vectors $S_1,....,S_K$ of size $S \in I\!R$ where $S \in I\!R^{D \times 1}$. The variable D specifies the feature dimensions of the CNN fc feature maps. Applying an aggregation function $\phi : S_1, S_2, ...., S_k \rightarrow f$, produced an individual cumulative feature map $f \in I\!R^{D \times 1}$. In other words the aggregation functions generated a condensed and robust single track-level feature representation of the whole track.

Before forwarding the feature vectors to the K-means clustering algorithm, the feature tracks were $l_2$ normalized, resulting in unit vectors $f\prime \leftarrow f/||f||_2$. The clustering algorithms merged the tracks of each unique identity to find the optimal quantity of clusters C - which

- *Steven Farrugia is an MSc Computing Science Student studying at RUG, E-mail: s.p.farrugia@student.rug.nl*
- *Amey Bhole is an MSc Computing Science Student studying at RUG, E-mail: a.bhole@student.rug.nl*

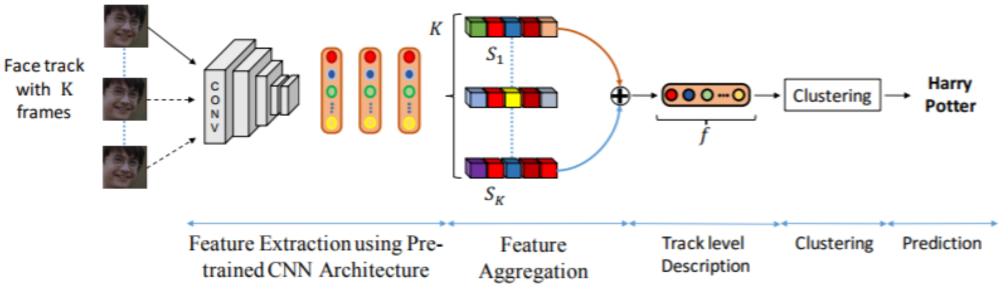


Fig. 1. Architecture defining the various stages from face tracking with K frames to the resultant cluster prediction results [20]

in this case refers to the number of main casts present in the episode being analyzed.

The proposed VDF, using average aggregation, coupled with K-means clustering, was compared with SIFT based Fished encoding ($VF^2$) [17]using two clustering algorithms; i) Hierarchical Agglomerative clustering with Negative Constraints (HAC-Neg) [23] and ii) Scene level clustering (TC) [23].

### 2.1.3 Measures to evaluate cluster quality

V. Sharma et. al evaluated the quality of the clusters using three measures widely adopted in video face clustering methods; i) Accuracy - calculated via a confusion matrix between the predicted and ground truth labels, ii) Weighted Clustering Purity (WCP) [23] - a metric which specifies the purity of each cluster computed as:

$$W = \frac{1}{N} \sum_{c \in C} n_c \bullet p_c \tag{1}$$

where N refers to the total sum of tracks in the video, $n_c$ the quantity of tracks present in the cluster $c\epsilon C$, and the corresponding purity $p_i$ calculated as the fraction of the largest number of tracks forming part of the identical $n_c$ label.,and iii) Operator Clicks Index (OCI-k) [6] given as OCI - k = C + E where E is the total number of wrongly clustered samples in C clusters. The OCI-k quantifies the clustering quality with respect to the quantity of clicks necessary to assign labels to all the face tracks for a given clustering.

### 2.1.4 Exploited datasets

Three video face clustering data-sets; Buffy the Vampire Slayer (BF), Big Bang Theory (BBT) and Notting Hill were used to evaluate the different aggregate functions. These datasets were evaluated on by a number of researchers using state of the arts methods; namely utilize distance metric learning (ULDML) [4], Hidden Markov Random Fields (HMRF) [26] and more, allowing for a comparison to take place.

Table 1. Data used during training and clustering evaluation

| Data-set | Main Casts | Season | Episode |
|---|---|---|---|
| BF0502 | 6 | 5 | 1-6 |
| BBT0101 | 5 | 1 | 1 |
| Notting Hill | 5 | 1 | 1 |

## 2.2 Clustering Millions of Faces by Identity

### 2.2.1 Face Representation

Using Kazemi and Sullivan's [11] ensemble of regression trees method capable of producing normalized images, C.Otto, D. Wang and A. Jain obtained a training set of 404,922 successfully aligned face images containing 10,533 distinct people from the CASIA-webface data-set. The resultant 100x100 normalized images were fed to the adopted very deep convolutional neural network architecture specified in [25], consisting of 10 convolution layers (which enhanced the non-linearity of the network) utilizing filters with a very small receptive field (3x3) [22] over all layers. The produced 320-dimensional vector generated by the final average pooling layer was used throughout the clustering experiments.

### 2.2.2 Proposed approximate Rank-Order Clustering

The Rank-Order clustering algorithm, proposed by Zhu et al. [27], was proven to be effective in the comparison of multiple clustering methods mentioned in 'An Efficient Approach for Clustering Face Images' [15].

The computation of nearest neighbor lists, using a cluster-to-cluster distance metric for each sample present within the data-set, raised a scalability issue leading to an overall cost of $0(n^2)$. Therefore to overcome this, the FLANN library implementation of the randomized k-d tree algorithm [14] was utilized in order to produce a small list of nearest neighbors. The adapted distance metric, taking only into account the top k neighbors, may be seen in the summation equation below:

$$d_m(a,b) = \sum_{i=1}^{min(O_a(b),k)} I_b(O_b(f_a(i)),k) \tag{2}$$

where $I_b(x,k)$ defines an indicator function with a resultant value of 0 if face x is present in face b's top k-nearest neighbors, else 1. $f_a(i)$ specifies the i-th face with respect to the neighbor list of face a, and $O_b(f_a(i))$ provides the rank of face $f_a(i)$ present in face b's neighbor list. Therefore the existence (as well as the absence) of an example in the predefined size k of the short list is more significant than the actual corresponding numerical rank.

A combined modified symmetric distance measure produced high values if two faces ranked high in their respective nearest neighbor and shared common neighbors (and vice-versa). The distance measure, specified as:

$$d_m(a,b) = \frac{d_m(a,b) + d_m(b,a)}{min(O_a(b), O_b(a))} \tag{3}$$

contributed towards more accurate clustering results. Runtime for the clustering procedure was diminished by 1) only calculating the distances between samples which share nearest neighbors and 2) executing only one round of merges of distinctive faces into clusters. This contributed towards a reduced complexity of $0(n)$.

### 2.2.3 Per-Cluster Quality Evaluation

Due to the data-set, likewise in practical situations, not containing the corresponding image labels, internal cluster quality measures [10] were utilized. To rank individual clusters, per-cluster quality measures were exploited on top of measures typically grouped as compactness or isolation. The coverage [2] measure was modified to only take into account notes present in the prevailing cluster in order to compute the per-cluster quality measure. Other quality measures such as modularization and intra-cluster connectivity to detect the presence or absence of edge between explicit vertices as well as a simple distance measures between edges in the k-NN graph (average inter-cluster distance and average intra-cluster distance) were also calculated and evaluated.

#### 2.2.4 Exploited datasets

Four unconstrained data-sets (seen in table 2) were used throughout the clustering experiments; namely the Casia-webface data-set, the Labeled Face in the Wild (LFW) and YouTubeFaces (YTF). A collection of 123M unlabeled web face images were also utilized and combined with labeled images to apply clustering evaluation over larger-scale data-sets.

Table 2. Data used during training and clustering evaluation

| Data-set | # of Face Images | # of subjects | Purpose |
|---|---|---|---|
| LFW | 13,233 | 5,749 | Training |
| YTF | 621,126 | 1,595 | Cluster Eval. |
| Webfaces | 123,654,141 | Unknown | Cluster Eval. |
| CASIA-webface [1] | 494,414 | 10,575 | Cluster Eval. |

### 2.3 Face Clustering: Representation and Pairwise Constraints

#### 2.3.1 Face Representation

C.Otto, D. Wang and A. Jain used the ”deep residual network” architecture proposed in [8] for face representation. The proposed ResNet model consisting of 18, 50 and 101- layer architectures were trained on different training set combinations of CASIA-Webface, VGG-Face and VGG-Deduplicated stated in table 3, where the the normalized face images were scaled according to the procedure proposed in [25] to 256 x 256 and 224 x 224 regions were randomly cropped during the training. The corresponding results were verified on the LFW dataset under BLUFR protocol and the best network architecture was used from the results produced for face representation.

#### 2.3.2 Proposed Conditional Pairwise Clustering

The Conditional Pairwise clustering method proposed by C.Otto, D. Wang and A. Jain is used to group face images based on their identity using pairwise similarity between images. In the proposed face clustering method the adjacency between all the pairs of faces is used as the variable to predict and look for an output that maximizes the joint posterior probability of these variables given their pairwise similarity. The problem is modelled as Conditional Random Field (CRF) and employs an non-standard Loopy Belief Propagation algorithm to achieve a valid adjacency matrix.
The Conditional Random Field model used to maximize the posterior probability is provided in the following equation:

$$p(Y|X) = \frac{1}{Z}\prod_{i<j}\psi_u(Y_{ij})\prod_{i<j<k}\psi_t(Y_{ij},Y_{ik},Y_{jk}) \tag{4}$$

where Z is the normalization factor, $Y_{i,j}$ is the binary variable which provides the information if the $X_i$, $X_y$ belong to the same cluster, i is the number of data points and X is a dataset. The unary association potential $\psi_u(Y_{ij})$ = $p(Y_{ij}|X_i,Xj)$ is the pairwise conditional distribution. The adjacency matrix is considered to be symmetric therefore only $Y_{ij}$ where i¡j are considered as variables. Unary action potential is the likelihood of a pair of data points belonging to the same class, therefore they applied a transformation to the cosine similarity between the deep representation of the two faces to achieve $\psi_u(Y_{ij})$ = 1. The triplet interaction potential used to constrain adjacency matrix to be valid is provided in the following equation:

$$\psi_t(Y_{ij},Y_{ik},Y_{jk}) = exp(-\alpha V((Y_{ij},Y_{ik},Y_{jk}))) \tag{5}$$

where V is the energy function which is 1 if and only if the triplet is inconsistent and 0 otherwise,

$$V((Y_{ij},Y_{ik},Y_{jk})) = (1-Y_{ij})Y_{ik}Y_{jk} + Y_{ij}(1-Y_{ik})Y_{jk} + Y_{ij}Y_{ik}(1-Y_{jk}) \tag{6}$$

Negative logarithm is taken on both sides of the equation 2 due to numerical issues which minimizes the energy function as follows:

$$E(Y,X) = \sum_{ij} D(Y_{ij} + \sum_{i<j<k} \alpha V((Y_{ij},Y_{ik},Y_{jk}))) \tag{7}$$

where $D(Y_{ij})$ = $-log\psi_u(Y_{ij})$ is the unary potential energy used to maximize the posterior probability. This model provides a graph structure where the output node $Y_{ij}$ contains N cliques of one pairwise clique with input pair as $X_i$, $X_j$ and N-1 triplet cliques consisting of $Y_{ij}$, $Y_{ik}$ and $Y_{jk}$. The graph factors and the potentials are used for minimizing the energy using a min-sum algorithm. The main steps of the algorithm are as follows:

1. Initialize all messages:

$$a^0_{ij}(Y_{ij}) = D(Y_{ij}) \tag{8}$$

In the above equation the message $a_{ij}$ is the accumulated energy for each state of $Y_{ij}$

2. At iteration t = 1,2,...T, update the messages as:

$$a^t_{ij}(Y_{ij}) = \sum_{k \in N^(t-1)(i,j)} min_{Y_{ik},Y_{jk}} a^{t-1}_{ij}(Y_{ij}) + a^{t-1}_{jk}(Y_{jk}) + \alpha V((Y_{ij},Y_{ik},Y_{jk})) \tag{9}$$

For each iteration t the messages are summed up from different factors

3. The final state of the variable is determined by:

$$Y_{ij} = \underset{Y_{ij}}{\operatorname{argmin}}\, a^T_{ij}(Y_{ij}) \tag{10}$$

The complexity of the algorithm is O($TNM^2$) where N is the number of data points, T is the number of iterations and M is the maximum degree of any data point in any iteration.

#### 2.3.3 Evaluation Measures

To evaluate the cluster quality of the algorithms Pairwise F-measure and BCubed F-measure were utilized, where both evaluation metrics compute F-score defined as the harmonic mean of Precision and Recall. In Pairwise F-measure, the Precision and Recall are calculated using the following formula:

$$PairwisePrecision = \frac{TP}{TP+FP} \tag{11}$$

$$PairwiseRecall = \frac{TP}{TP+FN} \tag{12}$$

where labels for all $\frac{1}{N}(N-1)$ pairs with N points in the dataset are used and therefore TP, FP, and FN can be defined as True Positive Pairs, False Positive Pairs and False Negative pairs

F-measure or F-score for both criteria is specified as:

$$F = \frac{2 \times Precision \times Recall}{Precision + Recall} \tag{13}$$

#### 2.3.4 Exploited datasets

Four distinct face image data-set were utilized for evaluation of the cluster quality and training of the deep residual network. CASIA-webface, VGG and VGG-Deduplicated were used in order to training of the deep residual network and LFW and IJB datasets were used for evaluation of the cluster quality . VGG-Deduplicated is a data-set version of VGG, consisting of examples after removing broken links, repeated mislabeled URLs and duplicate images which reduced the size of the data-set to 1.7 million face images as given in table 3.

Table 3. Data used during training and clustering evaluation

| Data-set | # of Face Images | # of subjects | Purpose |
|---|---|---|---|
| CASIA-webface [2] | 494,414 | 10,575 | Training |
| VGG | 2.6 million | 2,622 | Training |
| VGG-Deduplicated | 1.7 million | 2,622 | Training |
| LFW | 13,233 | 5,749 | Cluster Eval. |
| IJB-B-1 | 1,437 | 32 | Cluster Eval. |
| IJB-B-2 | 2,566 | 64 | Cluster Eval. |
| IJB-B-3 | 4,793 | 128 | Cluster Eval. |
| IJB-B-4 | 11,186 | 256 | Cluster Eval. |
| IJB-B-5 | 19,583 | 512 | Cluster Eval. |
| IJB-B-6 | 37,653 | 1,204 | Cluster Eval. |
| IJB-B-7 | 68,714 | 1,870 | Cluster Eval. |

## 3 Results

In this section we state and highlight the results obtained by each paper, as well as other methods which the researchers compared their work with.

### 3.1 Results for 'A simple and Effective Technique for Face Clustering in TV series

The number of tracks produced was 2.5 to 3 times the amount generated by other compared research papers. This was due to the use of the tracking-by-detection method with a particle filter.

Evaluating the clustering accuracy attained when using the aggregation functions in order to fuse the VDF fc7 features on the three face clustering datasets, Element-wise Average (Avg) proved to be the leading choice; obtaining 87.46%, 99.26% and 89.62% on the BF0502, Notting Hill and BBT0101, respectfully. This in turn was a substantial improvement over Element-wise Maximum (Max), and more so on the unsatisfactory Element-wise Multiplication (Mul).

Comparing the superior average aggregation of VDF features coupled with K-means produced competitive as well as effective WCP and OCI-k measures when compared to SIFT based Fisher encoding $(VF^2)$ with HAC-Neg and TC. As a baseline, the same amount of clusters defined in [23] was used. The produced results, over season 5 episodes 01-06 with respect to the OCI-k was on average approximately 30 above the results obtained by HAC-Neg and TC, over all episodes. The produced WCP ranged between 0.075 and 0.025 below that attained by HAC and TC.

### 3.2 Clustering Millions of Faces by Identity

Using k-means as a baseline, the proposed approximate rank-order clustering method is initially compared with Matlab r2015a renditions of k-means and spectral clustering, as well as the original rank-order they specified on the small LFW dataset. Due to the nature of k-means and spectral clustering requiring a pre-specified number of clusters, the algorithms were executed on a number of cluster values. As may be seen from table 4, the optimal F-measure for both k-means and spectral was attained on a low number of pre-defined clusters, 100 and 200 respectively. Assessing on a cluster value similar to the true number of identities, k-Means produced a relatively poor 0.07 F-measure metric when compared to 0.8 by 0.87 achieved by the original and altered Rank-Order algorithms.

Computation time ranged immensely; with Spectral taking exceedingly long in contrast with the other methods on 13,233 face images.

Contrasting the performance (F-measure and Run-time) of three nearest neighbor algorithms; i) Computing all pairwise comparisons ii) Chen et al. [19] and iii) Randomized k-d Tree [9] on LFW as well as LFW + 1M unlabeled images from the Webfaces dataset, Randomized k-d Tree was the superior method. The aforementioned attained F-measure was distinctively higher than the other two methods, even more so when the unlabeled images were added to the LFW dataset. While Brute force run-time was relatively close to that obtained by Randomized k-d Tree on the LFW dataset, there was a run-time improvement by a factor of 120 on the LFW + 1M unlabeled images.

Table 4. Comparison of results obtained for clustering algorithms with respect clusters created, F-measure and Run-time on the LFW dataset

| Clustering Algorithm | # Clusters | # F-measure | Run-time |
|---|---|---|---|
| k-Means | 100 | 0.36 | 00:00:16 |
| k-Means | 6,508 | 0.07 | 04:58:49 |
| Spectral | 200 | 0.20 | 00:11:18 |
| Rank-Order | 6,591 | 0.80 | 00:00:18 |
| Approx. Rank-Order (Proposed) | 6,508 | 0.87 | 00:00:08 |

The FLANN implementation of randomized k-d tree algorithm was then assessed on LFW with 1M and 5M additional unlabeled images using three different search size strategies; i) Fixed, ii) linear and iii) logarithmic increase. A fixed search size produced a mediocre 0.13 F-measure, more than 5 times less than that obtained when using a linear increase. The linear increase was found to be optimal when implemented on the LFW + 5M data-set but the computation time for the approximation method increased from $O(nlog(n))$ to $O(n^2)$.

Using a distributed memory variation of the randomized k-d tree nearest neighbor approximation [14] on a large scale face data-set, clustering results were evaluated on the LFW data-set amalgamated with 1M, 5M, 10M, 30M and 123M unlabeled background images (seen in 5). Since the tree structure had to be entirely loaded into memory, storage and computation of $0(n^2)$ became unfeasible on a single machine. Therefore data-sets were split into disjoint sets with 1 core assigned per 1M unlabelled images. Run-time took over 2 weeks, with the workload spread across 123 nodes on the LFW + 123M data-set. The corresponding F-measure of 0.27 was obtained, which is still substantially better than a random result.

Table 5. Large scale clustering results utilizing the proposed Approximate Rank-Order clustering method on top of which k-d tree nearest neighbor approximation was used.

| Dataset | F-measure | # Clusters | Cores | Run-time |
|---|---|---|---|---|
| LFW | 0.87 | 1,463 | 1 | 00:00:19 |
| LFW+1M | 0.79 | 94,740 | 1 | 01:03:25 |
| LFW+5M | 0.67 | 445,880 | 5 | 06:28:42 |
| LFW+10M | 0.56 | 933,278 | 10 | 12:11:33 |
| LFW+30M | 0.42 | 2,800,202 | 30 | 30:44:58 |
| LFW+123M | 0.27 | 10,619,853 | 123 | 289:04:53 |

### 3.3 Face Clustering: Representation and Pairwise Constraints

#### 3.3.1 Face Representation

For face representation the ResNet model is trained on different training sets and the performance was verified on LFW data set using the BLUFR protocol. Results from [24] are used as the baseline. Training a 50-layer network on a combination of the CASIA-Webface and VGG-Deduplicated provides best verification rate at 0.1% FAR as 92.22%. However training the same dataset on a 101-layer network reduced the observed error but did not improve the performance. The CASIA-webface dataset was trained on a 18-layer residual network as a proof of concept to attain the baseline given in [24]. A 50-layer ResNet was trained on the complete VGG-Face dataset as well as CAISA dataset and achieved a verification rate at 0.1% FAR as 89.74%. Moreover traning a 50-layer ResNet with fully pre-activated network on individual datasets CASIA and VGG didnt improve the results significantly.

### 3.3.2 Face Clustering

The proposed clustering algorithm ConPaC was evaluated on two unconstrained LFW and IJB-B where the threshold $\tau = 0.7$ is the parameter used through out the experiment which controls the prior knowledge and clustering results. The ConPaC algorithm was implemented in C++ and evaluated on Intel Xeon CPU clocked at 2.90GHz using 24 cores. The proposed algorithm is compared to the following benchmark clustering algorithm:(1) K-means, (2) Spectral Clustering [7] (3) Rank-order clustering [3] and Approx. Rank-order clustering [18]. Implementation of K-means was carried out on MATLAB R2016a and third-party MATLAB implementation of and Spectral clustering was used while evaluating the cluster quality.

Table 6. Comparison of the F-measures of ConPaC and other clustering algorithm on LFW dataset

| Algorithm | Pairwise | BCubed | # Clusters | Run-time |
|---|---|---|---|---|
| K-means | 0.098 | 0.680 | 5,749 | 00:04:08 |
| K-means | 0.359 | 0.460 | 500 | 00:00:14 |
| Spectral | 0.033 | 0.559 | 5,749 | 01:00:56 |
| Spectral | 0.257 | 0.249 | 75 | 14:37:14 |
| Rank-Order | 0.813 | 0.891 | 5,699 | 00:00:33 |
| Approx. Rank-Order | 0.861 | 0.875 | 6,801 | 00:00:12 |
| ConPaC (proposed) | 0.965 | 0.922 | 6,352 | 00:00:39 |

Table 6 represents the results obtained on the LFW dataset where the number of clusters is required as an input parameter for K-means and spectral clustering. Therefore the performance for K-means and spectral clustering was evaluated with true number of clusters C, C = 5,749 after which the clustering was repeated for several different values and the best performance were reported. As one may see from table 6, the proposed ConPaC algorithm outperforms all other clustering algorithms with Pairwise F-measure of 0.965, BCubed F-measure of 0.922 and run time of 39 secs.

Table 7. Comparison of the pairwise F-measures of ConPaC and other clustering algorithm on IJB-B-7 dataset

| Algorithm | Pairwise | BCubed | # Clusters | Run-time |
|---|---|---|---|---|
| K-means | 0.313 | 0.398 | 1000 | 00:06:56 |
| Spectral | 0.208 | 0.335 | 500 | 01:34:40 |
| Rank-Order | 0.005 | 0.267 | 4,084 | 01:12:25 |
| Approx. Rank-Order | 0.315 | 0.317 | 31,218 | 00:00:73 |
| ConPaC (proposed) | 0.239 | 0.429 | 23,119 | 02:53:58 |

Table 7 represents the results for IJB-B-7 dataset with 1870 subjects. As the number of identities increase the F-score for all the algorithms decreases. However ConPaC algorithm performs better than other competing algorithms where the number of identities are less.

## 4 Discussion

In this section we compare and contrast the similarities, dissimilarities and possible improvements of the three paper approaches. For the purposes of this discussion as well as readability, paper 1 is ”A simple and Effective for Face Clustering in TV Series”, paper 2 is ”Clustering Millions of Faces by Identity and paper 3 is ”Face Clustering: Representation and Pairwise Constraints”.

### 4.1 Dataset training for face representation

All three papers took different approaches with respect to the training and selection of the face representation model. Paper 1 made use of the full 2.6 million VGG dataset, paper 3 use the Casia Webface dataset and paper 2 used a combination of both the VGG and Casia webface datasets. Paper 2 also carried out pre-processing of the VGG datasets(cleaning of the datasets), which involved removing duplicate images, broken links and images that overlapped in both datasets. This pre-processing step was not carried out by paper 1 and 3 on the datasets, which may have improved feature representation results.

### 4.2 Selecting best architecture for face representation

Paper 1 selected the VGF model for face representation. After obtaining track level feature vectors, they compared 3 aggregation methods in order to obtain a single compressed track. From the three aggregation methods, defined in section 2.1.2, element-wise-average was found to be the superior aggregation function, producing highly discriminative features for the whole track. Paper 2 compared multiple architectures of ResNet trained on different combinations of the three datasets (VGG de-duplicated, VGG and Casia Webface) and selected the model which produced the best FAR results. Paper 3 utilized a Deep CNN with 10-Convolution layers (ConvNet). Paper 2 took the best approach in order to find the best architecture for face representation, by comparing a number of different layers as well as different combinations of the datasets; which paper 1 and 3 did not address.

### 4.3 Clustering Methods & Evaluation

The compressed track level feature vector, produced in paper 1, following element wise average was then fed to k-means clustering. When compared to HAC-Neg and TC clustering methods, using the same feature vector, K-means not only produced competitive but is also is less complex and requires less parameters.

Main-casts often change their appearance from one episode to another; change of clothes, wigs, fake moustache, make-up and so on. Essentially, not conjoining multiple episodes may have aided the evaluation results obtained by not taking into account these previously mentioned possible factors. In real-world applications of face clustering, multiple videos over different time-periods or images may need to be analyzed which in turn may be comprised of similar aspects, not to mention possible aging and facial scabs,scars and wounds. Another possible improvement would be the use of clustering methods which are capable of inferring the correct amount of clusters without any human intervention (i.e. Number of main-casts in the videos are not pre-determined).

Table 8. Comparison of the F-measures of ConPaC and Approximate rank ordering algorithm on LFW dataet

| Algorithm | Dataset | F-measure | # Clusters | Run-time |
|---|---|---|---|---|
| ConPaC (full graph) | LFW | 0.965 | 6,352 | 00:00:39 |
| ConPac (k-d tree) | LFW | 0.964 | 5,927 | 00:03:06 |
| Approx. Rank-Order (k-d tree) | LFW | 0.87 | 6,508 | 00:00:08 |
| Approx Rank-Order (Brute Force) | LFW | 0.72 | 6,508 | 00:00:12 |
| Approx Rank-Order (Approx k-NN graph) | LFW | 0.69 | 6,508 | 00:26:36 |
| ConPac (k-d tree) | LFW + 1M | 0.809 | Unknown | 04:24:09 |
| Approx. Rank Order ( k-d tree) | LFW + 1M | 0.79 | Unknown | 01:03:25 |
| Approx. Rank Order ( Brute-Force) | LFW + 1M | 0.49 | Unknown | 14:18:24 |
| Approx. Rank Order (Approx k-NN graph) | LFW + 1M | 0.41 | Unknown | 01:06:58 |

The results for the two proposed algorithms in paper 2 and 3 can be compared since both papers used the same dataset for testing the performance of the algorithms. As seen from Table 8 both variations of ConPac have nearly the same F-measure results, however the number of clusters produced varied on LFW. When comparing the three nearest neighbour algorithms( Brute Force, K-d tree and approximate k-NN graph) used in paper 3 for approximate rank ordering, the approximate K-d tree obtained the best F-measure result. This in comparison with ConPac applied on the LFW and LFW + 1Million face images, produced less accurate results. Run-time as well as computational complexity was much higher for the ConPac algorithm, requiring 24 CPUs as opposed to 1 CPU required for Approximate rank order with k-d tree for Nearest neighbors. ConPaC also predicted a larger number of clusters than the true number of clusters present in the face image dataset due to the algorithm considering a larger number of outliers as singleton clusters.

When paper 3 evaluated the Approximate rank ordering algorithm on a dataset of LFW + 123Million face images, clustering accuracy was shown to progressively decay as the data-set size increased. Scalability issues were addressed by splitting the LFW + 123 Million face images into disjoint indexed subsets across a network with 1 core per 1 million images. C. Otto et al. obtained an F-measure of 0.27, which although low was considerably better than a random result (which is almost 0).

A lower overall F-measure was attained when applying the same method on the YTF data-set when compared to the F-measure on the similar sized LFW + 1M. This may primarily be attributed to the lower image quality. More clusters were generated than the identities present, with relatively pure clusters. Based on the recall measure, the algorithm was successful in group frames in videos but lacked success in grouping identities across videos. As stated by C. Otto, this may mainly be due to the 200 nearest neighbors of each face including frames of the same video. Therefore as suggested, clustering the identities within the video first to ensure distinguishable identities following clustering again on a reduced data to reinforce the per-video identities across various videos is recommended.

Large scale datasets, in comparison to the 123-Million face images, were not used by paper 1 and 2 for evaluation of the respective face clustering approaches.

## 5 Conclusion

In this review paper we have highlighted three different approaches researchers have proposed or carried out in order to cluster face images according to identity on a large scale, with some as large as 123 Million images. From the three papers reviewed we conclude 3 factors which play an important role in face clustering for real world applications in order to achieve optimal results; pre-processing of training data, scalability such that the algorithm may be applied to a large dataset - which helps identify any computational as well as time complexity present in the algorithm and finding a suitable Deep learning architectures through tests.

Paper 1 was able to highlight the importance of feature representation by obtaining competitive results when compared with state of the art algorithms. Paper 2 carried out an in-depth pre-processing on the training data which helped in optimizing the face representation results as well as proposed a new algorithm. However when compared with paper 3, which also proposed a new algorithm, slightly better results were obtained but at a cost of computational complexity and time. The algorithm proposed in paper 3 was evaluated was the only algorithm applied on a large scale dataset containing 123 Million face images which appropriately represents a real world scenario.

## Acknowledgements

The authors wish to thank Andre Sobieki.

## References

[1] Top 20 facebook statistics - updated february 2018, Feb 2018.

[2] U. Brandes, M. Gaertler, and D. Wagner. Experiments on graph clustering algorithms. In *European Symposium on Algorithms*, pages 568–579. Springer, 2003.

[3] D. Chen, X. Cao, L. Wang, F. Wen, and J. Sun. Bayesian face revisited: A joint formulation. In *European Conference on Computer Vision*, pages 566–579. Springer, 2012.

[4] R. G. Cinbis, J. Verbeek, and C. Schmid. Unsupervised metric learning for face identification in tv video. In *Computer Vision (ICCV), 2011 IEEE International Conference on*, pages 1559–1566. IEEE, 2011.

[5] E. Ghaleb, M. Tapaswi, Z. Al-Halah, H. K. Ekenel, and R. Stiefelhagen. Accio: A data set for face track retrieval in movies across age. In *Proceedings of the 5th ACM on International Conference on Multimedia Retrieval*, pages 455–458. ACM, 2015.

[6] M. Guillaumin, J. Verbeek, and C. Schmid. Is that you? metric learning approaches for face identification. In *Computer Vision, 2009 IEEE 12th international conference on*, pages 498–505. IEEE, 2009.

[7] J. Han, J. Pei, and M. Kamber. *Data mining: concepts and techniques*. Elsevier, 2011.

[8] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[9] A. K. Jain. Data clustering: 50 years beyond k-means. *Pattern recognition letters*, 31(8):651–666, 2010.

[10] A. K. Jain and R. C. Dubes. Algorithms for clustering data. 1988.

[11] V. Kazemi and J. Sullivan. One millisecond face alignment with an ensemble of regression trees. In *2014 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1867–1874, June 2014.

[12] J. C. Klontz and A. K. Jain. A case study of automated face recognition: The boston marathon bombings suspects. *Computer*, 46(11):91–94, Nov 2013.

[13] O. M. Parkhi, A. Vedaldi, and A. Zisserman. Deep face recognition. 1:41.1–41.12, 01 2015.

[14] M. Muja and D. G. Lowe. Scalable nearest neighbor algorithms for high dimensional data. *IEEE transactions on pattern analysis and machine intelligence*, 36(11):2227–2240, 2014.

[15] C. Otto, B. Klare, and A. K. Jain. An efficient approach for clustering face images. In *Biometrics (ICB), 2015 International Conference on*, pages 243–250. IEEE, 2015.

[16] C. Otto, D. Wang, and A. K. Jain. Clustering millions of faces by identity. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 40(2):289–303, Feb 2018.

[17] O. M. Parkhi, K. Simonyan, A. Vedaldi, and A. Zisserman. A compact and discriminative face track descriptor. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1693–1700, 2014.

[18] S. Sankaranarayanan, A. Alavi, C. D. Castillo, and R. Chellappa. Triplet probabilistic embedding for face verification and clustering. In *Biometrics Theory, Applications and Systems (BTAS), 2016 IEEE 8th International Conference on*, pages 1–8. IEEE, 2016.

[19] F. Schroff, D. Kalenichenko, and J. Philbin. Facenet: A unified embedding for face recognition and clustering. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 815–823, 2015.

[20] V. Sharma, M. S. Sarfraz, and R. Stiefelhagen. A simple and effective technique for face clustering in tv series. 2017.

[21] Y. Shi, C. Otto, and A. K. Jain. Face clustering: Representation and pairwise constraints. *CoRR*, abs/1706.05067, 2017.

[22] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *CoRR*, abs/1409.1556, 2014.

[23] M. Tapaswi, O. M. Parkhi, E. Rahtu, E. Sommerlade, R. Stiefelhagen, and A. Zisserman. Total cluster: A person agnostic clustering method for broadcast videos. In *Proceedings of the 2014 Indian Conference on Computer Vision Graphics and Image Processing*, page 7. ACM, 2014.

[24] L. Tran, X. Yin, and X. Liu. Disentangled representation learning gan for pose-invariant face recognition. In *CVPR*, volume 3, page 7, 2017.

[25] D. Wang, C. Otto, and A. K. Jain. Face search at scale. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 39(6):1122–1136, June 2017.

[26] B. Wu, Y. Zhang, B.-G. Hu, and Q. Ji. Constrained clustering and its application to face clustering in videos. In *Computer Vision and Pattern Recognition (CVPR), 2013 IEEE Conference on*, pages 3507–3514. IEEE, 2013.

[27] C. Zhu, F. Wen, and J. Sun. A rank-order distance based clustering algorithm for face tagging. In *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*, pages 481–488. IEEE, 2011.

# An Overview of Detection of Faint Astronomical Sources

Gert-Jan van Ginkel, Mark Helmus

**Abstract**— Automatic detection of faint astronomical sources has long been a challenge using the most common tools. The most used tool for this purpose is SExtractor. SExtractor was introduced in 1996 and object detection has advanced rapidly since. In this paper we look at two newer methods: NoiseChisel and MTObjects. Both these methods use a different approach at solving the same problem. With these faint astronomical sources often being very similar to the background it is difficult to distinguish the less dense regions from noise. NoiseChisel makes use of image processing, along with the signal-noise ratio in order to detect nebulous objects. MTObjects makes use of image processing as well, but uses statistical methods in order to detect faint sources. Both NoiseChisel and MTObjects prove to be a very good alternative to SExtractor and greatly outperform the latter on a specific faint source.

**Index Terms**—Faint astronomical sources, SExtractor, NoiseChisel, MTObjects.

## 1 Introduction

Detection of faint astronomical sources can be a difficult process. Since these astronomical objects (e.g. galaxies) can have a profound effect on the behavior of galaxy clusters, it is important to detect these objects effectively. Manual detection is not feasible given the huge amount of data generated by telescopes; therefore this process should ideally be automated.

Three methods for automatically extracting these astronomical sources are discussed in Section 2. The first method, SExtractor [3], is a commonly used program to extract sources from telescope images. NoiseChisel [1] is the second method; this method use a different noise-based approach to find objects. Finally MTObjects [9] uses statistical methods to extract the background and determine significant objects.

Both NoiseChisel and MTObjects are compared to SExtractor in Section 3 since SExtractor is commonly used as a baseline for new methods. Finally, a short conclusion is given in Section 4.

## 2 Existing methods

In this section three algorithms are described. The first, SExtractor, is a commonly used program by astronomers and often used as a baseline for new methods. The second method is the NoiseChisel method. This method makes use of image processing techniques and then applies segmentation to detect the nebulous objects. The third and last algorithm described is MTObjects, which uses a predominantly statistical approach.

### 2.1 SExtractor

The first and most commonly used method is SExtractor [3]. SExtractor is able to detect, deblend, measure and classify sources from images. SExtractor follows a number of steps to achieve these goals. These steps are as follows: background estimation, detection, deblending merged objects, filtering the detections, photometry and the separation of stars and galaxies. The photometry and separation steps are not discussed, since they are not relevant to our research which focuses on the detection and segmentation of objects. The other steps are further discussed in detail in their corresponding subsections.

#### 2.1.1 Background estimation

The first step in extracting sources is estimating the background of the input image. Assuming the image is the result of a certain background with the light of the objects added to it, this step estimates that background. This background estimation is done using a method similar to the one described by Da Costa [5]. Instead of thresholding on the entire image, the image is divided into a grid and thresholding is applied locally.

This local thresholding is performed in a number of steps. First, the local histogram is clipped using $\sigma$-clipping. This iteratively clips the histogram by removing everything outside $\alpha\sigma$ of the mean of the histogram. Here $\sigma$ is the current standard deviation of the histogram and $\alpha$ a control variable. This clipping is repeated until the estimated background converges to $\pm 3\sigma$ of the median.

An estimate for the crowdedness can be obtained by seeing how much the $\sigma$ of the background estimate has changed from the original value. A cell is considered crowded if this change is more than 20% and uncrowded otherwise. Finally, if the cell is uncrowded, the background is estimated as the mean of the clipped histogram. If the cell is crowded, though, the background is estimated as $2.5 \times \mathit{median} - 1.5 \times \mathit{mean}$.

With the local background estimated for all cells, a median filter can be applied. This median filter will mitigate the effect of bright objects increasing the brightness of the local background. Finally, the background map is created by a bilinear interpolation of the cells.

#### 2.1.2 Detection

Now that the background has been found and removed, the stars and galaxies have to be detected. The detection has to be able to deal with objects of any shape, so one of the most basic and generic methods is used: thresholding. This thresholding is done in combination with a filter on the image.

First, a filter is applied to the image with a given convolution mask. The shape of this convolution mask has a great impact on what kind of objects SExtractor will find. One suggestion for faint sources is a Point Spread Function. This function has a peak in the center of the domain and quickly falls off further away from the center. Next, thresholding is applied to this convolution mask and 8-connected components are extracted. These components are then used as the base objects that will be used by the rest of the algorithm.

#### 2.1.3 Deblending of merged objects

Since the detection step can give one single component for two different objects it is necessary to separate these merged objects. One example of merged objects are two bright stars very close to each other; these are so close the thresholding will group them into one object. Deblending is done in two steps: finding the peaks in the grouped objects, followed by reallocating the pixels to their respective peak.

Finding the peaks starts with making a model of the light distribution of the object. The intensities of the image are discretized into 30 levels which are spaced on a logarithmic scale. The range of this discretization is from the original extraction threshold (the one chosen in the detection phase) to the maximum value of the object. This discretization of the light level is stored in a tree; an example of such a tree is shown in Figure 1. Next the algorithm starts at the branches

- *Gert-Jan van Ginkel is a MSc student at the University of Groningen.*
- *Mark Helmus is a MSc student at the University of Groningen.*

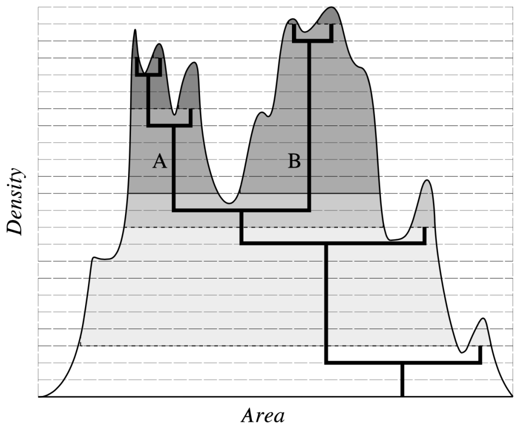


Fig. 1: Schematic taken from [3] of the deblending method used in SExtractor.

of the tree and splits the object at intensity level $t_i$ if the following conditions are met:

1. The area of this branch above $t_i$ is greater than a given percentage of the full area of the entire object.
2. The first condition has to be true for at least one other branch on intensity level $t_i$.

Giving lower values to the percentage of condition 1 results in more frequent splitting whereas higher values will give the opposite effect. The second condition ensures that splitting the object will actually give two objects that satisfy the first condition. As a result this will prevent small peaks (e.g. the rightmost peak in the example from Figure 1) from registering as an object. Peak A and B on the other hand are registered as two peaks, since their area is large enough for the threshold.

After the different objects and their cores are determined, the surrounding pixels need to be reassigned to their respective object. Gaussian distributions that are fitted to the profile of the objects are used to determine the likelihood of any pixel belonging to a specific object. The pixel is then allocated with the given likelihood as probability to that object. This means multiple runs with the same input image can give slightly different results.

### 2.1.4 Filtering the detections

Spread out galaxies and nebula have a shallow profile – they can be represented as a Gaussian distribution with a high standard deviation. The algorithm has a problem with these shallow profiles because the background is estimated locally. This means that for galaxies and nebula that are very spread out it is possible that the nebula itself is estimated as the background, which in turn causes noise to be detected as an object.

The solution for this problem is verifying whether the detection would still have been made regardless of the brightness of the neighborhood. The contribution of the neighbors is estimated by building a Gaussian model of their profiles, after which it is subtracted from the profile of the object. If the profile still satisfies the detection threshold (as described in Section 2.1.2) it is kept, and discarded otherwise.

## 2.2 NoiseChisel

NoiseChisel was proposed by Akhlaghi et al. [1] NoiseChisel is a non-parametric algorithm, which has grown out of the definition of the Sky value (a concept in any measurement which is significantly affected by noise). The pixel units in NoiseChisel are assumed to be in photon-counts (counting the number of arriving photons in each pixel), which are referred to as 'counts'. The algorithm assumes that the total count of a detected object , O, is desired. Then, for each pixel the total count can be written as:

- $O_{ij}$, the contribution from the target object
- $D_{ij}$, the contribution from other detected objects
- $U_{ij}$, undetected objects or the fainter undetected regions of bright objects
- $C_{ij}$, the cosmic rays
- $B_{ij}$, the background count, defined as the count if none of the others exists on that pixel

Thus, the total count in pixel $T_{ij}$ can be written as:

$$T_{ij} = O_{ij} + D_{ij} + U_{ij} + C_{ij} + B_{ij} \tag{1}$$

By definition, $D_{ij}$ is detected and the algorithm assumes that it is correctly subtracted. Therefore, $D_{ij}$ is set to zero. Moreover, there are methods to detect and remove cosmic rays [11], setting $C_{ij}$ to zero. Furthermore, $D_{ij}$ and $U_{ij}$ are correlated as both directly depend on the detection algorithm and its input parameters. Leaving the observed Sky value for pixel $S_{ij}$ now defined as:

$$S_{ij} = B_{ij} + U_{ij} \tag{2}$$

As the detection process becomes more accurate ($U_{ij} \to 0$), the Sky value will tend toward the background value ($S_{ij} \to B_{ij}$). This means that $S_{ij}$ depends on the detection process, as $B_{ij}$ is a pixel in the image. Over a group of pixels, this equation translates to the average of undetected pixels. Using the definition of Sky, the object count in the data can be calculated with:

$$T_{ij} = S_{ij} + O_{ij} \Leftrightarrow O_{ij} = T_{ij} - S_{ij} \tag{3}$$

This means that the more accurately $S_{ij}$ is measured, the more accurately the count of the target object can be calculated. Using the definition, the Sky value is only correctly found when all the detected objects ($D_{ij}$ and $C_{ij}$) have been removed from the data.

The algorithm detects nebulous objects in the following manner:

1. Initialization
2. Convolve the input image
3. Apply thresholding to the smoothed image
4. Eroding regions below the threshold
5. Separate objects by opening
6. Estimate false detections and remove them
7. Reverse erosion of the fainter regions of true detections by applying dilation

These steps are outlined in the next sections.

### 2.2.1 Initialization

For initialization, the algorithm uses an approach which is independent of the Sky value to arrive at an initial detection. Individual initial detections are classified as true or false based on their signal-to-noise ratio ($S/N$), using the surrounding undetected regions as a basis. This classification is then used as the initialization.

### 2.2.2 Convolution kernel

After initialization, the algorithm starts by performing a convolution on the input image. It is used to maximize the ratio of an object's peak signal to the local noise level [4]. Using convolution, high spatial frequencies are removed, smoothening the image. After convolving, neighboring pixels receive an intermediate value, if they have high variance (because of noise). Thus, by smoothing the image using convolution, the signal-to-noise ratio increases. Noise limits the ability to detect a contiguous region if the algorithm would not apply convolution. Therefore, it is necessary to apply convolution in order to maximize the detection ability. The algorithm receives the number of neighboring pixels, along with the type of kernel used are given as input of the algorithm. Determining the number of neighboring pixels is important, since using a wider convolution kernel has the following consequences (the more neighboring pixels used, the wider the kernel):

- Shapes of the object tend to the shape of the kernel.
- The wider the convolution kernel, the more the dynamic range decreases, thus making it harder to separate fainter parts of brighter objects become harder.
- Compact objects, i.e. objects that are not wide relative to the convolution kernel, will be lost.

The authors advise to use a sharp convolution kernel in order to find the faint object pixels.

### 2.2.3 Thresholding

After convolving, thresholding is applied such that the algorithm will work with a binary image from thereon. Thresholding in NoiseChisel is defined using the cumulative distribution of the pixels in the convolved image. In order to ensure that the threshold is lower than the Sky value, the quantile has to be less than the median. After thresholding, the pixels having a value above the threshold are assigned a black pixel and pixels below the threshold are assigned a white pixel.

### 2.2.4 Erosion

After thresholding, erosion is applied to the binary image. Erosion is an operation applied in morphological image processing [7]. It erodes the foreground (the black pixels), where it is implied that all foreground pixels neighboring a background pixel are carved off the foreground and changed to a background pixel (the white pixels). The neighborhood of a pixel is defined based on the structuring element. NoiseChisel uses a 4-connected structuring element to erode the foreground. Eroding the foreground expands the holes and connects them to each other, separating the augmented regions.

### 2.2.5 Opening

After eroding the image, opening is applied to the eroded image. An opening is, as erosion, an operation applied in morphological image processing. An opening is defined as an erosion followed by a dilation. With dilation, when a background pixel touches a foreground pixel, it will be changed to a black pixel (when dilating the foreground). In the algorithm, the opening is used to separate the foreground into separately connected regions.

### 2.2.6 Defining and removing false detections

After opening, the foreground consists of separately connected objects, which are now defined as initial detections. The first detections divide the pixels into two sets, the undetected sky, $R_s$ and the detected regions, $R_d$. An assumption of the algorithm is that no significant contribution from a detectable object exists in $R_s$. Furthermore, some detections in $R_d$ contain a signal, while others are noise pixels, i.e. false detections. To identify the false detections in $R_d$, a second detection process is applied independently to both pixels of $R_s$ and $R_d$. Detections in $R_s$provides the scale on which certainty of detections in $R_d$ is defined. The second threshold is defined by the average count $S_a$ and the standard deviation of the non-detected regions $\sigma_s$ of $R_s$. $S_a$ is not the final Sky value, as some of the false detections have been removed. As holes arise most probably due to noise, filling holes by first eroding them and then opening them enhances the ability to identify a faint signal. In order to remove thin connections between thicker regions, one level of 4 connected opening is applied. Furthermore, let $F$ be the average count in each pseudo detection and let $N$ be its area. The parameter used to quantify the definition of a false detection is the total $S/N$ of each pseudo-detection (a combination of its total count and area). For each pseudo-detection, $S/N_t$ can be written as:

$$S/N_t = \frac{NF - NS_a}{\sqrt{NF + N\sigma_s^2}} = \frac{\sqrt{N}(F - S_a)}{\sqrt{F + \sigma_s^2}} \tag{4}$$

This distribution of detections in $R_s$ provides a scale that can be used to select or reject a given detection in $R_d$, the level of certainty for true detections is defined by user input.

### 2.2.7 Final dilation

The final step is to restore the pixel layers of each remaining initial detection that were removed by erosion. Dilation has been explained in the subsection describing the Opening.

### 2.2.8 Segmentation

As galaxies have no clear cutoff, it can occur that nearby objects on the image will be detected as one region. Two approaches to deal with this problem are to perform sub-detection or finding substructure: deblending and segmentation. As stated previously, deblending identifies the contribution in each pixel from different blended sources. In deblending, each pixel can belong to more than one object. Deblending is the more realistic approach: the sum of the light profiles of multiple overlapping objects specifies the final pixel value. However, deblending will require parametric analysis and fitting algorithms. Segmentation is a low-level measurement, assigning each pixel to the subcomponent contributing to it. In order to segment each detection into sub-components, clumps and objects are defined. We will outline in the next sections how clumps and objects are defined.

### 2.2.9 Finding 'true' clumps

Pixels at a local maximum are defined as pixels whose value is larger than their 8 connected neighbors. The 8 connected region around each local maximum whose pixels all have a lower count than the peak is defined as a clump. The clump can thus not be larger than 9 pixels. After finding clumps, the algorithm identifies false clumps. In order to do so, it will make use off the signal-noise ratio of each clump. The signal-noise ratio of each clump can be written as:

$$S/N = \frac{N_iF_i - N_iF_0}{\sqrt{N_iF_i + N_iF_0}} = \frac{\sqrt{N_i}(F_i - F_0)}{\sqrt{F_i + F_0}} \tag{5}$$

, where $F_i$ and $F_0$ are defined as average count in a clump and and the average count on all pixels surrounding it respectively. Furthermore, $N_i$ is the total area of number of pixels inside a clump. This distribution is used to find the S/N threshold to accept or reject a clump over the detected object. The 99th quantile of the distribution is used to find the $S/N$ threshold to accept or reject a clump over the detected object. This threshold is independent regarding the object in which the potential clumps are embedded. As layering is not necessary anymore and the counting of clumps are not compared to the object, clumps in various galaxies can now be objectively compared with each other, after finding the 'true' clumps.

### 2.2.10 Object segmentation

Since low thresholds are used, objects close enough to each other on the image will be blended in one detection. If a detected region has none or only one clump, it is considered to be only object. If more than one clump is detected, the clumps grow until a certain threshold of the input image is reached. This threshold is defined by the Sky value and standard deviation. Growing clumps works very similar to the algorithm for finding the clumps, where the only difference is that no new labels are added. If a pixel has no labeled neighbors, it is kept in a queue to be checked on a next loop. The loop continues until no new pixels can be labeled. Objects are defined based on the average $S/N$ of the river pixels (where river pixels act as separators of each clump) between the grown clumps. Two grown clumps are defined as separate objects if the river between them is below a user-defined signal-to-noise ratio $S/N$. If it is larger, then the connection between the grown clumps is too strong to be regarded as separate objects. Then, the grown clumps are considered as parts of one object. Let $F_{ij}$ be the average count on the river between the clumps $i$ and $j$. The average $S/N$ of the river between these two clumps is defined as:

$$S/N_{ij} = \frac{F_{ij} - S}{\sqrt{F_{ij} + \sigma_s}} \tag{6}$$

. If $S/N_{ij} < bordersn$, where *bordersn* is input given by the user, then these two clumps are considered separate objects and if not, they are defined as if they belong to the same object. $S/N_{ij}$ is calculated for all clumps of each detection. Finally, all the clumps that connected to each other within a detection are given one label.

## 2.3 MTObjects

MTObjects was proposed by Teeninga et al. [9] [10] as an alternative to SExtractor. MTObjects was designed with faint astronomical sources in mind, which is something SExtractor struggles with. MTObjects is based on a constant background estimation, a hierarchical representation of the image and using statistical analysis to identify objects in the image. This section describes each of these principles in more detail.

### 2.3.1 Background estimation

Teeninga et al. suggested a new and improved way to estimate the background of a tile in [10]. This new method uses a constant background estimation whereas SExtractor uses a local estimate of the background. The image is first segmented into tiles, where each tile is classified as a flat or non-flat tile. A tile is flat if the values could be drawn from a single Gaussian distribution. In order to determine with rejection rate $\alpha$ whether the tile is flat two tests have to be performed on the tile:

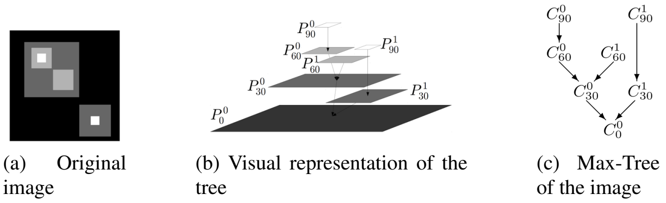
1. A normality test using the D'Agostino-Pearson $K^2$-statistic [6]
2. $t$-test for equal means on different segments of the tile

The normality test with rejection rate tests whether the tile could be created with a single Gaussian distribution. This test is performed with rejection rate $\alpha_1 = 1-(1-\alpha)^{1/2}$. This test, however, does not take into account the positions of the values. This means that gradients could be classified as being drawn from a single Gaussian, while they are not. The additional $t$-test is performed to alleviate this problem. The $t$-tests are performed on vertically and horizontally split sections of the image with rejection rate $\alpha_2 = 1-(1-\alpha)^{1/4}$. The tile is classified as non-flat if the normality test is rejected or any of the additional $t$-tests' null hypothesis are rejected.

The final background estimation is then calculated by taking the mean and variance of the flat tiles in the image. The size $w_i$ of the flat tiles is iteratively increased by $w_{i+1} = 2w_i$ as long as there is at least one flat tile in the image for that tile size.

### 2.3.2 Creating a Max-Tree of the image

The next step is creating a Max-Tree of the image. The estimated background computed in the previous step is first subtracted and negative components are set to 0. This Max-Tree can be constructed by creating a tree of connected components in the image. The leaves in this tree correspond to the local maxima of the image and the root of the tree corresponds to the lowest value in the image, i.e. the estimated background of the image. Figure shows how the original image is transformed into a hierarchical structure. The lowest level in the tree corresponds to the background level.



(a) Original image

(b) Visual representation of the tree

(c) Max-Tree of the image

Fig. 2: Image and visual representation of its Max-Tree. Taken from [9]

### 2.3.3 Finding significant nodes

The Max-Tree constructed in the previous step can be used to find significant nodes. Teeninga et al. provide four different tests to determine whether a node is significant or not. Let $P_{anc}$ be the closest significant ancestor of $P$, $f(x)$ be the value of pixel $x$ and $O$ be the noiseless image. Additionally there are two power definitions for a node $P$:

$$\text{power}(P) := \sum_{x\in P} (f(x) - f(\text{parent}(P)))^2 \tag{7}$$

$$\text{power}_{alt}(P) := \sum_{x\in P} (f(x) - f(P_{anc}))^2 \tag{8}$$

Four significance tests are designed in order to find the significant nodes in the Max-Tree of the image.

**1. Power given area of the node** The first significance test uses the power definition from Equation 7. A node is significant if we can show that $O(x) > f(P_{anc})$; which translates to the original noiseless image having a higher power than the local background. $P$ will be significant if $H_{power}$ is rejected for some pixels $x \in \text{parent}(P)$. $H_{power}$ is defined as follows:

$$H_{power} = \forall x \in P : O(x) \leq f(\text{parent}(P)) \tag{9}$$

for pixels $x \in \text{parent}(P)$. It is assumed that the distribution of the power attribute scaled by the variance $\sigma$ for noise nodes follows a $\chi$ distribution. Furthermore, for a random pixel $x$ in $P$, $\frac{(f(x)-f(parent(P)))}{\sigma}$ is $\chi$ distributed with 1 degree of freedom.

The hypothesis is rejected if $\text{power}(P)/\sigma^2 > \texttt{inverse}\chi^2\texttt{CDF}(\alpha,\texttt{area})$, where $\texttt{inverse}\chi^2\texttt{CDF}(\alpha,\texttt{area})$ returns the rejection boundary given by the $\chi$ cumulative distribution, for a significance level $\alpha$.

**2. Alternative power given area and distance** The exact distribution of $power_{alt}$ is not known. Therefore, the distribution is obtained by Monte Carlo simulation. It generates Gaussian noise images, with mean and variance equal to the estimates. Thereafter, a number $n$ of independent values is generated.

The following hypothesis is used for this significance test:

$$H_{powerAlt} = \forall x \in P : O(x) \leq f(P_{anc}) \tag{10}$$

In order to make the significance level more constant for each node, independently of tree height, the ancestor node is used in the computation of the power attribute. The algorithm defines $distance(P) = f(P) - f(P_{anc})$. Furthermore, $X$ is a random set of $area(P) - 1$ values, drawn from a truncated normal distribution, having a minimum value of $distance(P)$. The variance $\sigma$ is set to $\sigma = \hat{\sigma}_{bg} + g^{-1} f(P_{anc})$. The function $inversePowerAltCDF(\alpha, area, d)$ returns the estimated rejection boundary for the power attribute. Hypothesis $H_{powerAlt}$ is rejected if $powerAlt(P)/\sigma > inversePowerAltCDF(\alpha, area, d)$, meaning that if the object image $O(x)$ at some pixels $x$ in $P$ is higher than $f(P_{anc})$, P is marked as significant. The minimum area of a significant node is 2 pixels. In the case that the boundary is not available for a $distance$, linear interpolation between rejection boundaries is used.

**3. Alternative power given area** This test is independent of the $distance$ measure. Using assumptions from significance test 2, $distance(P)$ has a truncated normal distribution with a minimum value of 0 (which is the same distribution as a random non-negative pixel value). The rejection boundary is then calculated using simulated noise images along with four-connectivity. Furthermore, using a different connectivity changes the rejection boundary.

**4. Alternative power given area, using a smoothing filter** This test is equal to significance test 3. The only difference is that the image is smoothed beforehand. Smoothing the image is used to reduce noise and to detect more objects. Thus, more objects are detected using this test. The same smoothing filter as in SExtractor is applied:

$$H = \frac{1}{16}\begin{bmatrix}1 & 2 & 1\\ 2 & 4 & 2\\ 1 & 2 & 1\end{bmatrix} \tag{11}$$

This filter is applied after the background subtraction and before setting the negative values to zero. After smoothing the image, pixel values are not independent anymore. Decision boundaries are then determined again by performing Monte Carlo simulations.

### 2.3.4 Utilizing significant nodes to find objects

Because the nodes that are marked significant could be overlapping each other (e.g. a significant node has a significant ancestor) they need to be separated into different objects. If a significant node $P$ has no significant ancestors it is automatically marked as an object. A significant node can also be marked as an object when $P_{anc}$ has another descendant that has a higher area than $P$.

Taking these markers as the final positions in the tree introduces a problem with noise. If the markers are set too close to the background level it is possible for the background noise to be included in the objects. Therefore the markers are moved up the tree (away from the background level) proportional to the standard deviation of the noise. This is done in such a manner that for every object marker $P$ the following holds true: $f(P_{final}) \geq f(P_{anc}) + \lambda\sigma^2_{background}$. Let $P_{final}$ be the final position of the marker and $\lambda$ the control parameter. High values for $\lambda$ will result in less noise, but could potentially remove too much of the extent of the source. Low values for $\lambda$ on the other hand will include too many noise pixels into the object. As a result a compromise of $\lambda = 0.5$ is recommended.

## 3 Differences and comparison

Although making a comparison between all methods would be desirable, we decided to only compare NoiseChisel and MTObjects to SExtractor. Since SExtractor is the standard and most widely used this is a solid benchmark to compare these methods to. All these comparisons are made with faint astronomical sources in mind – some differences do not have a big impact whereas some will. The methods are compared on two aspects: background estimation and detection. All three methods perform some kind of background estimation which is then used to separate the objects from the background. Finally the detection of objects from the segmented image is compared.

### 3.1 NoiseChisel

SExtractor and NoiseChisel have a lot of similarities. As will be shown below, they both apply background estimation and detection, although each applies it in their own manner.

#### 3.1.1 Background estimation

Where SExtractor uses a local estimation of the background, NoiseChisel deals with estimating the background in a different manner. First, the NoiseChisel algorithm starts with applying kernel convolution to the image, which smoothens the image. Then, a very low threshold is applied to the smoothed image. Using this threshold, the regions below the threshold are expanded by eroding the regions that are above the threshold. After applying the erosion, an opening is applied to the remaining regions. Now, all pixels that were not part of the remaining regions, have become the background. Like MTObjects, NoiseChisel uses a more global approximation of the background. This has the same advantage of preventing large faint objects increasing the local background, thus preventing detection of objects around that area.

#### 3.1.2 Detection

Both methods use a detection phase. Where SExtractor uses an initial detection phase and refines that detection later using two more phases, NoiseChisel first detects regions by applying image processing techniques, and then creating binary images, where the pixel values of this image depend on a threshold. After this phase, NoiseChisel applies segmentation to detect sub regions in regions, by dividing clumps where feasible, which is the second phase of detecting. An example where NoiseChisel performs better than SExtractor is for surfaces with low brightness areas. Since SExtractor is designed for using high threshold values, when reliable peaks are found (using the high threshold), each pixel is assigned to a peak. The SExtractor assumes that all peaks follow a Gaussian profile over the whole image. In reality, this is not (always) the case. Figure 3 compares the results of SExtractor and NoiseChisel on different images. While SExtractor tends to leave out a lot of the objects, NoiseChisel has a tendency to keep too much. One could possibly argue SExtractor performs better on this specific dataset since SExtractor gives a clearer outline.

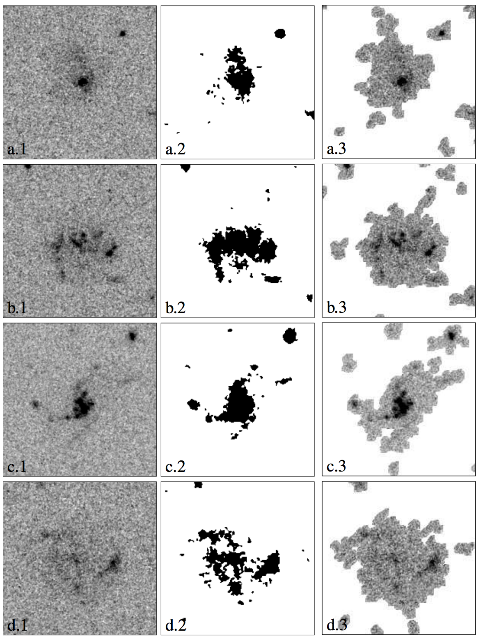


Fig. 3: Comparison of SExtractor and NoiseChisel on a galaxy. The first column shows the input image, the second shows the output using SExtractor and the third shows output using NoiseChisel. As can be seen, NoiseChisel is able to identify the nebulous objects better than SExtractor. Taken from [1]

### 3.2 SExtractor and MTObjects

SExtractor and MTObjects have a lot in common – the general structure of their execution is very close. Some parts could potentially even be swapped among them to improve their performance.

#### 3.2.1 Background estimation

The background estimation for both methods can be compared really well – both use background estimation with the same goal in mind, but using different methods. SExtractor uses a local estimation of the background whereas MTObjects computes this background globally which results in a constant background. Although it is possible to enter a constant background in SExtractor [2] this is not an automated process like it is in MTObjects.

Figure 4 shows the background estimation by SExtractor and MTObjects for each column in an image and the respective average column value. It becomes apparent that using the adaptive background estimation has a strong correlation with the input image. This poses a big problem for large faint sources that show up as a wider but lower peak. MTObjects on the other hand does not suffer from the same problem – the background is constant over the entire image.

Figure 5 shows the average background estimation for all images in the dataset used by [10]. Not only does it display varying estimates

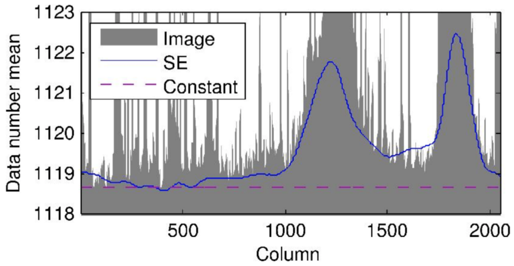


Fig. 4: Difference in background estimation for SExtractor and MTObjects for a single image [8].

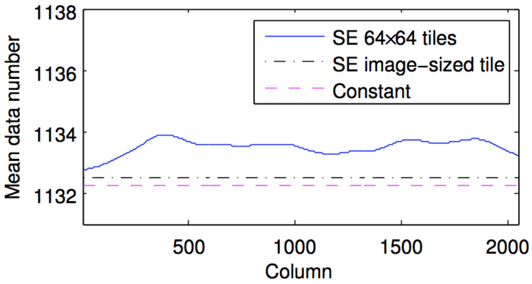
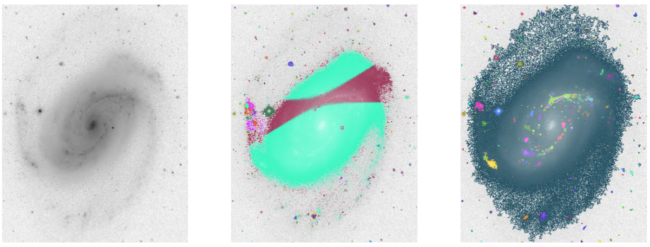


Fig. 5: Difference in background estimation for SExtractor and MTObjects [10]

of the background, but it also shows that the constant background estimate is lower than the background estimation from SExtractor. Having this lower background estimate is positive for detecting faint sources; less of the original image is subtracted, so more signal is 'left' for the detection phase to process.

### 3.2.2 Detection

Both methods use a detection phase. Where SExtractor uses an initial detection phase and refines that detection later using two more phases, MTObjects instead marks nodes on a Max-Tree of the image as objects. As a result of using this Max-Tree, MTObjects is able to better detect nested objects. SExtractor on the other hand does not have the capability to detect these. This gives a huge advantage to MTObjects for faint astronomical sources like wide and shallow galaxies, since it is possible to detect multiple brighter objects nested in that galaxy. A comparison of the detection of a galaxy is shown in Figure 6



(a) Original image (b) SExtractor (c) MTObjects

Fig. 6: Comparison of SExtractor and MTObjects on a galaxy. Taken from [9]

MTObjects correctly detects the entire galaxy with some nested objects of interest, while SExtractor fails to detect the galaxy as one region. Additionally MTObjects has a much wider range of detection: even the distant and faint arms of the galaxy are correctly identified and assigned to the galaxy.

## 4 Conclusion

We have given an overview of the methods SExtractor, NoiseChisel and MTObjects. Furthermore, the differences among the algorithms are shown along with their unique traits.

NoiseChisel proves to be more effective in identifying nebulous objects by applying multiple image processing techniques. Furthermore, as the method is non-parametric, the algorithm does not assume a certain statistical distribution and is able to outperform SExtractor in certain instances. It does, however, fail to give a clear outline of the detected objects.

MTObjects also proves to be a lot more effective on the faint sources. The added capabilities for nested objects is very useful for the outer ranges of astronomical objects as well. Overall, MTObjects is preferred over SExtractor for these kind of objects.

While both NoiseChisel and MTObjects are compared to SExtractor they are not compared to each other. Looking at the results when compared to SExtractor, however, it appears that MTObjects gives a clearer result that still includes low density areas. Meanwhile, NoiseChisel tends to keep too much information around the objects. Future work could thus contain qualitative and quantitative comparisons of these two methods. Additionally, testing these methods on different kinds of datasets could give an insight in the generalization capabilities of these algorithms.

## Acknowledgements

The authors wish to thank Nadia Hartsuiker, Roel Visser and expert review M.H.F. Wilkinson for their feedback on the paper.

## References

[1] M. Akhlaghi and T. Ichikawa. Noise-based detection and segmentation of nebulous objects. *The Astrophysical Journal Supplement Series*, 220(1):1, 2015.

[2] E. Bertin. *SExtractor User's manual*. Institut dAstrophysique & Observatoire de Paris.

[3] E. Bertin and S. Arnouts. Sextractor: Software for source extraction. *Astronomy and Astrophysics Supplement Series*, 117(2):393–404, 1996.

[4] A. Bijaoui and M. Dantel. Shilt's method: Application to stellar photometry with the electronic camera. *Astron. and Astrophys*, 6:51–59, 1970.

[5] G. Da Costa. Basic photometry techniques. In *Astronomical CCD Observing and Reduction Techniques*, volume 23, page 90, 1992.

[6] R. B. D'agostino, A. Belanger, and R. B. D'Agostino Jr. A suggestion for using powerful and informative tests of normality. *The American Statistician*, 44(4):316–321, 1990.

[7] J. Starck and F. Murtagh. Astronomical image and data analysis. 2006.

[8] P. Teeninga, U. Moschini, S. Trager, and M. Wilkinson. Bi-variate statistical attribute filtering: a tool for robust detection of faint objects. *power*, 2:1, 2013.

[9] P. Teeninga, U. Moschini, S. C. Trager, and M. H. Wilkinson. Statistical attribute filtering to detect faint extended astronomical sources. *Mathematical Morphology-Theory and Applications*, 1(1), 2016.

[10] P. Teeninga, U. Moschini, S. C. Trager, and M. H. F. Wilkinson. Improving background estimation for faint astronomical object detection. In *2015 IEEE International Conference on Image Processing (ICIP)*, pages 1046–1050, Sept 2015.

[11] P. G. van Dokkum. Cosmic-ray rejection by laplacian edge detection. *Astronomical Society of the Pacific*, 113(789):1420–1427, 2001.

# Classification of Imbalanced Simulated Data Sets

Joey Antonisse and Siebert Elhorst

**Abstract**— In the modern world classification becomes more and more important, it can potentially save lives when classifying harmful diseases. Most data sets don't have the luxury of containing a lot of data, this sometimes creates a majority and minority class also known as an imbalanced data set. Standard classifiers are not able to cope with the class imbalance, these classifiers are biased towards the majority class. Examples of these methods are: K nearest neighbour (KNN), C4.5 and Support vector machines (SVM). These methods are commonly described as being a "weak" classifier. The combination of these problems is usually referred to as the Class Cover Problem (CCP). In order to solve class imbalance multiple methods are presented, these methods are usually build up on these "weak" classifiers. The most important one being Class Cover Catch Digraphs (CCCD) and furthermore resampling, SMOTE+ENN and a few smaller ones. In the experiment it was clear that these methods performed much better in comparison to the "weak" classifiers. Where CCCD performed much better than all the other classifiers since it has much less overlapping in comparison to the other methods. Summarizing, imbalanced data can be solved using several methods. Where CCCD being the best solution that was covered in this paper.

**Index Terms**—Class Cover Catch Digraphs, Class Cover Problem, Resampling, SMOTE+ENN, K nearest neighbour, C4.5, Support vector machines

## 1 Introduction

In the modern world classification becomes more and more important, it can potentially save lives when classifying harmful diseases. This makes correct classification crucial and essential. In classification the goal is to improve performance and achieve the highest possible accuracy. Most examples are applied to balanced data sets containing a lot of data that makes classification straight forward. However many real world problems don't have the luxury of containing a balanced data set or struggle to find large amount of data, this makes it hard to achieve high accuracy.

This paper focuses primarily on the two-class imbalance of a data set, where one class is more present (more occurrences) then the other class (see, e.g., Chawla [6] et al. (2004) and Lpez [7] al. (2013)). The two-class imbalance is a problem that occurs in different areas such as: medicine, fraud, anomaly and fault detection. In these areas the majority class hinders the classification of the minority class, because the majority class has far more observations than the minority class. To give an example of this phenomenon: The paper by Mazurowski (Mazurowski [5] et al., 2008) describes this problem very well. In this paper 5% of the patients (data points) were classified as having cancer and 95% as not having cancer. This is a typical example of class imbalance, where one class (Not having cancer) gets dominated by the other class (Having cancer), causing incorrect classification.

Most classification methods that are used within machine learning are biased towards a majority class, such as: K nearest neighbour (Fix and Hodges [3], 1989; Cover and Hart [2], 1967) and Support vector machines. These methods assume that the data set is balanced and contain the same amount of observations for both classes. But as described above this is not always the case. Due to this bias, these classification methods are keen to make incorrect classifications. To give an example: When looking at figure 1 it's visible that the circle class has more observations than the star class. When a high value of K is selected (20+), the odds of being classified as the circle class is almost 100% even if the new data point is a star class.

The whole bias and imbalanced data problem can be summarized as the Class Cover Problem (CCP). This paper provides approaches on how to solve the CCP as well as selecting a so called "target class" which represents a region in a data set. Some of the solutions to the CCP problem are: Class Cover Catch Digraphs (CCCD) classifier, re-sampling and Synthetic Minority Over-sampling Technique (SMOTE). All methods will be discussed, but the main topic is the CCCD classifier which tries to achieve a balance between the amount of positive data points and the negative data points, causing the learner to be unbiased to a certain class. Using these different methods with different simulations, the best performer/classifier with respect to the imbalanced data set is found. The results illustrate the performance of different strategies under certain circumstances attempting to achieve the highest performance.

## 2 Methods

There are a lot of methods that are not able to cope with imbalanced data because these methods are biased towards the majority class. Figure 1 highlights an imbalanced data set on a 2d-plane. When K nearest neighbour (*k*-NN) (*k*-NN is considered a weak classifier against imbalanced data) is applied to a data set as in figure 1 it will most likely miss classify some data points. This is mainly because *k*-NN searches for the neighbours of a new data point that are closest to this new data point. But since the star class has significant fewer data points it has a higher chance to select the negative circle class. This makes *k*-NN vulnerable to biased data and causing a lot of miss classifications.

This problem is defined as the Class Cover Problem (CCP). CCP means that all elements that are covered are specific to a class, with this cover a simple classifier can be constructed. The CCP can be a box, square, rectangle or disc (circle) problem depending on the classifications of the data in the subset. The term "Class Cover" corresponds to the problem of finding the area where the class is represented as itself. This corresponds to the area where no non-target points are located but only the target class. The cover or region can then act as a basic classifier for new points. The difference in class size will seriously impair the classifier because of the imbalance problem, hence the relation between the class imbalance problem and the CCP. The CCP will be larger because the area for which the minority class is in, will probably be significantly smaller than that of the majority.

This section explains different methods that partly/fully solve

- *Joey Antonisse, E-mail: J.P.Antonisse@live.nl.*
- *Siebert Elhorst, E-mail: siebert.reinier@gmail.com.*

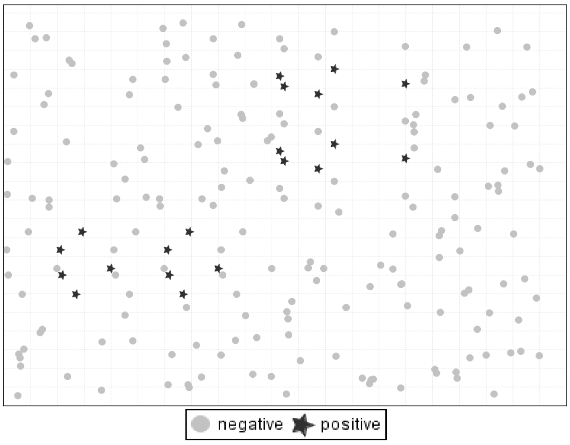


Fig. 1. Imbalanced Data Example

the issue of imbalanced data. Most methods described below are methods that were primarily build up on a weak classifier. But are modified in such a way that these methods are able to tackle the problem of imbalance. The methods were changed on either: 1) data level or 2) algorithmic level. Where data level modifies the data during training and algorithmic level adds a decision rule that is specific for that data set. There are also methods that are changed on both data level and algorithmic level the so called ”hybrid” methods, such as: resampling methods, cost-sensitive methods and ensemble methods. The methods described below are either modified versions of a weak classifier or a whole new classifier.

## 2.1 Class Cover Catch Digraphs (CCCD)

The Class Cover Catch Digraphs is a solution to the Class Cover Problem (CCP). The CCCD method/algorithm searches for a small number of sets covering data points from one class without any data points from the other class, see figure: 2, 3 and 4. CCCD uses different shapes to cover a class such as: rectangles, discs, balls and circles etc. In these images CCCD creates circles that covers one class, with a center of the ball representing it. Figure 2 is an example of one cover that contains only one class. Figure 3 shows the union of the dominating sets, of all the balls. Each individual ball has its own radius depending on the data points. The distance measure for CCCD is euclidean distance. Sometimes the CCCD is not able to cover all the data points, this issue is called an improper cover and the other way around is called a proper cover.

CCCD has many variations such as Pure CCCDs (P-CCCDs) and Random Walk CCCDs (RW-CCCDs). These different variations are described below

### 2.1.1 Pure CCCD

Pure CCCDs as it states is a *”pure”* algorithm, which means that there are no non-target classification points in the cover region. A problem arises when there are outliers or there is noise in the data set due to the nature of the disc size in CCCDs. To overcome the problem of outliers or noise, Random Walk CCCDs is introduced. Random Walk allows non-target points in the cover region of the CCCDs. This is further explained in Random Walk subsection. The ratio of target versus non-target points should be kept as high as possible, such that the sacrifice of containing a non-target point is compensated with multiple target classifications.
Pure CCCDs (P-CCCDs) work by assigning points a disc with radius $r(x)$; where the radius value is the distance from point $x$ to nearest point $y \in Y_m$ and point $y$ is a point from the non-target class. The size of the disc covers a region where all points in the region are from the target class. Any point in the target class that is closer than radius $r(x)$, will be in the subset of the cover, the calculation for radius $r(x)$ may be seen in Equation 1 (marchette).

$$r(x) := (1-\tau)d(x,l(x)) + \tau d(x,u(x))\imath \tag{1}$$

where

$$u(x) := argmin_{z \in Y_m} d(x,y) \tag{2}$$

and

$$l(x) := argmax_{z \in X_n}\{d(x,z) : d(x,z) < d(x,u(x))\} \tag{3}$$

The P-CCCDs class cover is pure of non-target members where there are no non-target points in the cover, hence the name ”pure”. This gives some problems when having noise or outliers in the non-target class, since the circle that goes around the point can not expand, because it would acquire non-target points denoted as the black dots, see Figure 2. The covering set is equal to the minimum dominating set which represents the cover with a minimum amount of samples.

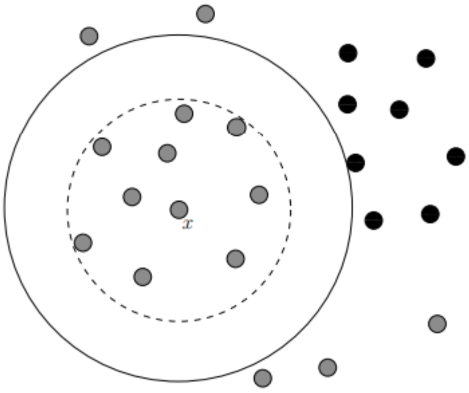


Fig. 2. *”The effect of $\tau ontheradiusr(x)$ of the target class point $x$) in a two-class setting. Grey and black points represent the target and non-target classes, respectively. The solid circle centered at $x$ is constructed with the radius associated with $\tau = 1$, the dashed circle is constructed with radius associated with $\tau = \varepsilon$ (DeVinney [4], 2003)”*, taken from [1].

Both algorithms reduce the amount of points in the majority class because the circle will try to catch as many target-class points as possible in an as dense area as possible. The selected center of the circle is a perfect representation for the neighbouring points, which is the goal of the CCCDs. By reducing the surrounding points but selecting the points that represent most surrounding points while the data set is kept intact known as prototype selection. The objective is to find a subset S to increase the classification performance while reducing the data set.

### 2.1.2 Random Walk CCCD

For the Random Walk CCCD the non-target class point is allowed in the target cover. If the radius r increases such as in Figure 2 the cover region can add more target class points but also acquires more non-target points. This can potentially reduce the chance of overfitting

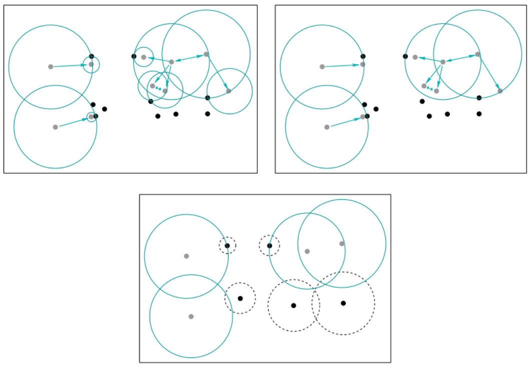


Fig. 3. *"An illustration of the CCCDs (with the grey points representing the points from the target class) in a two-class setting. Presented in top left are all covering balls and the digraph D = (V, A) and in top right are the balls that constitute a class cover for the target class and are centered at points which are the elements of the dominating set S Ł V (D). In the bottom panel, we present the dominating sets of both classes and their associated balls which establish the class covers. The class cover of grey points is the union of solid circles, and that of black points is the union of dashed circles."* Taken from [1]

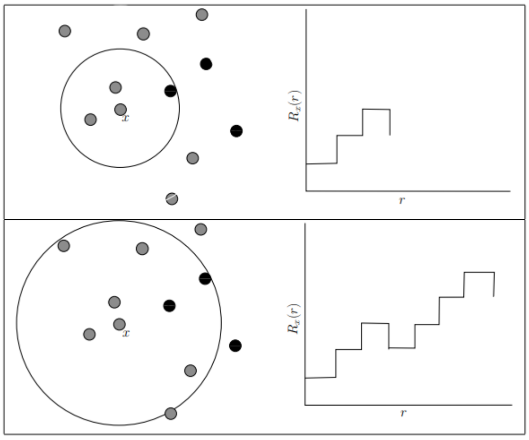


Fig. 4. *"Two snapshots of $R_x(r)$ associated with the ball $B_x$ centered at $x$ for $m = n$"*, taken from [1].

by allowing non-target in the area and target points outside. Figure 4 shows the amount of points vs radius (non-target: -1, target: +1) where a non-target reduces the step and causes a dip.

The pure CCCD will always produce a proper cover, where there are no non-target classes in the region, when compared to the Random Walk non-target class in the classification region are allowed. CCCDs can be transformed to work with multiple classes and classifications. The target class in the multi-class problem is the same as before, however the non-target class is now the union of all non-target classes together. With the union of these non-target sets we can apply the CCCDs with added classes.
An additional penalty function $P_x(r)$ is needed to make the circle still as small as possible with more target class points. The new updated range $r(x)$ is shown in Equation 4.

$$r(x) := argmax_{r \in \{d(x,z):z \in X_n Y_m\}} R_x(r) - P_x(r) \tag{4}$$

Even with the penalty function, $P_x(r) = 0$ in practice works sufficient as stated by De Vinney [4] (2003). The results of the RW-CCCDs versus the P-CCCDs can be viewed in Figure 5. The Difference between the two algorithms and the limitations of the Pure CCCDs have different effects.

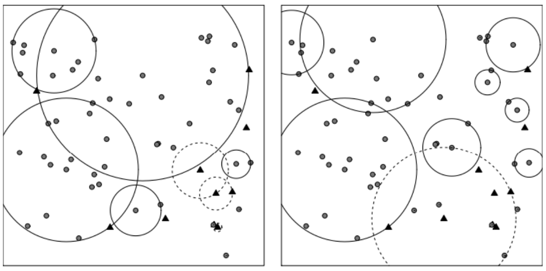


Fig. 5. *"An illustration of the covering balls associated with majority and minority P-CCCD (left) and the corresponding RW-CCCDs ( = 0.0001) (right) of an imbalanced data set in a two-class setting where majority and minority class points are represented by grey dots and black triangles, respectively."* Taken from [1].

### 2.2 Resampling

Resampling is one of the more common used method to tackle the effect of class imbalance in the classification process. With resampling parts of the data is either removed or added to the data set, with this method it's possible to decrease the amount of data points for the majority class and getting an equal amount of data points for both classes. Removing parts of the majority class is also known as downsizing and creating new data points for the minority class is called oversampling.

Resampling is a good method in order to reduce the class imbalance but it also has it weaknesses. The biggest weakness is that it removes data and usually that is not the best solution because then the data gets biased. The same problem occurs when adding new data points, these data points must be based on data points that were already present, which also makes the data biased.

### 2.3 SMOTE+ENN

Synthetic Minority Over-sampling Technique (SMOTE) generates artificial points for the minority class in order to balance the data set. Edited Nearest Neighbours (ENN) is an undersampling method that removes noisy data points, however ENN does not necessarily balances the data set like SMOTE. The goal of ENN is to increase performance of weak classifiers.

Adding SMOTE and ENN to work together we can oversample the training data with SMOTE by generating artificial data points between the original data points until the data set is balanced. After the balancing, ENN algorithm can clean the data by removing noisy data points, this is possible by checking if all points and their $k$ neighbours are labelled as the class of the selected point. This point is removed from the set if the classified label is not the same. SMOTE+ENN is used as the resampling scheme in the modification of the classifiers that are compared later in the paper.

### 2.4 Other methods

Another family of methods, namely cost-sensitive learning methods, has originated from real life: the cost of misclassifying a minority and a majority class member is usually not the same. Most of the time, the minority class has a higher misclassification cost in comparison to the majority class. This may be addressed by using decision trees which can be modified to take these costs into account.

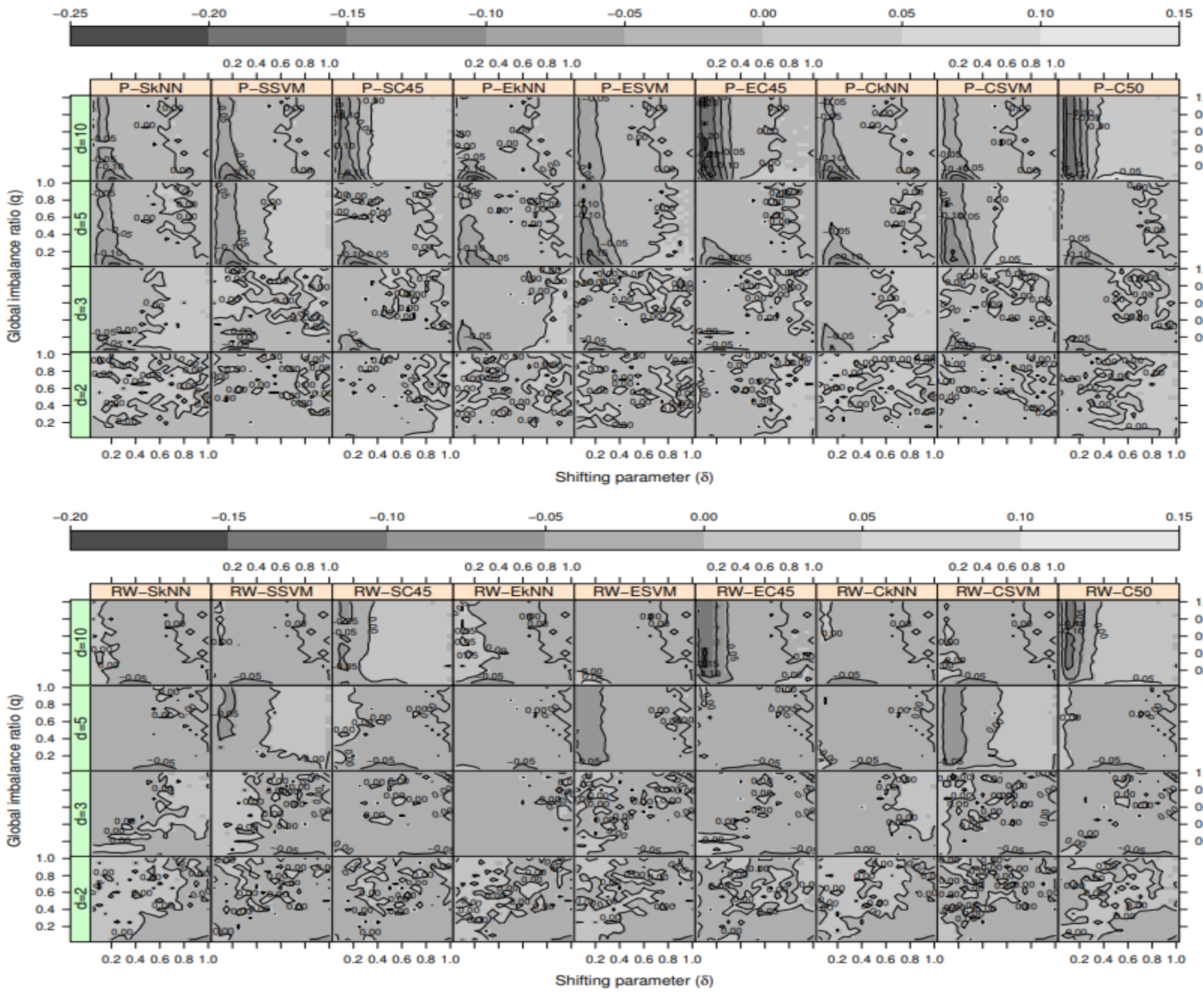


Fig. 6. *"Differences between the AUCs of CCCD and other classifiers. P-CCCD in (top) and RW-CCCD in (bottom) "*, taken from [1].

## 3 Experiments

This section discusses the experiments that were performed in order to test the different methods. The methods are applied to both simulations and real world problems.

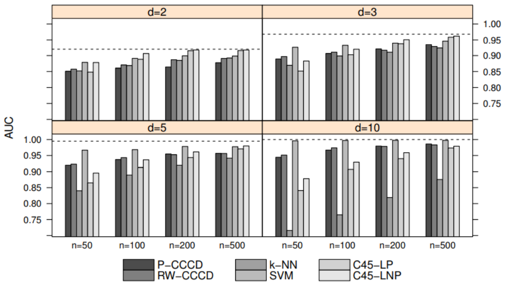


Fig. 7. CCRs in the two-class setting, $FX = U(0,1)^d$ and $FY = U(0.3,0.7)^d$ under various simulation settings, with d = 2, 3, 5, 10 and equal class sizes m = n = 50, 100, 200, 500. taken from [1].

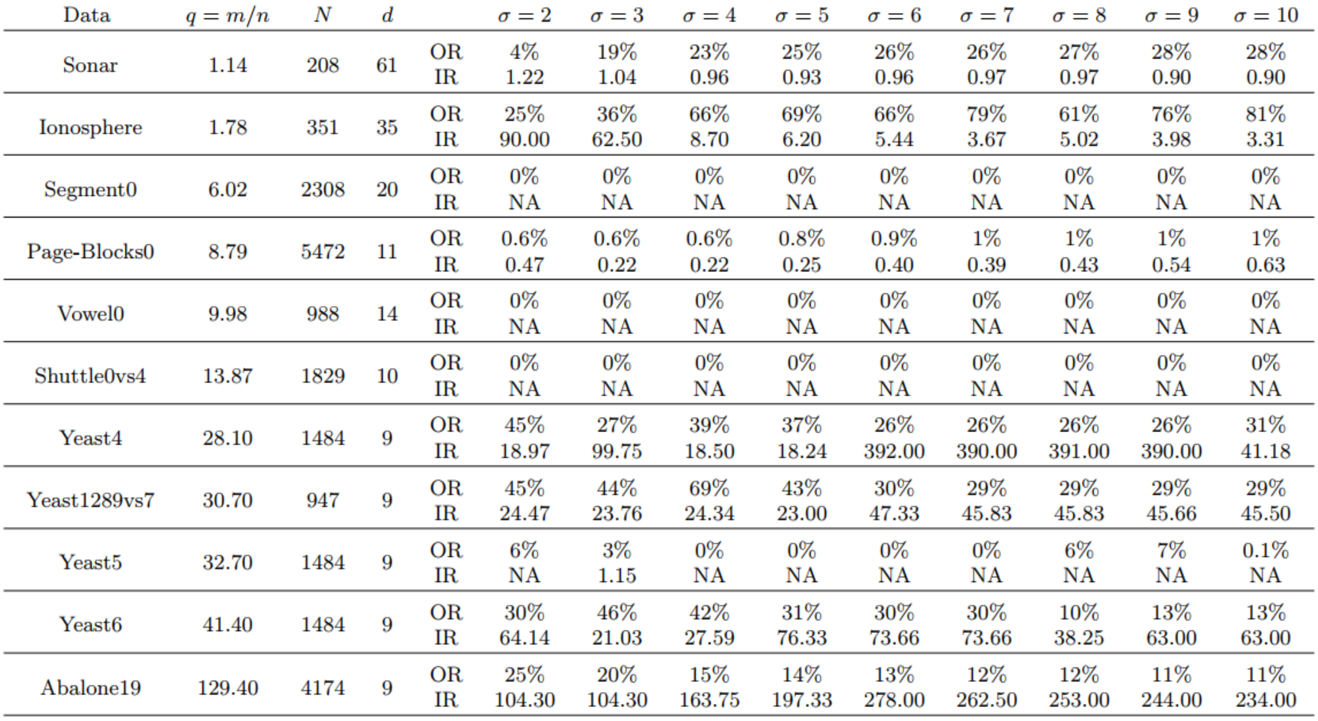
### 3.1 Simulations

This section describes the results of the simulations that were done. With the Area Under Curve (AUC) the performance of the classifier on imbalanced data sets is determined. A toy data set is generated with two classes from a multivariate uniform distribution. The observation size increases in different replications. Each replication the AUC is observed with 100 observations from each class. The data set is simulated until all classifiers achieve an error below 0.0005.

In the simulation the *k*-NN does not require any training time, therefore the entire training set can be used. However for the CCCD and Support vector machine (SVM) [8] the training set is reduced to the minimum dominating set. The reduction of a set is only when the classes are imbalanced because the resampling reduces the size to find a balance.

#### 3.1.1 Weak Classifiers

Both pure CCCD (P-CCCD) and Random Walk CCCD (RW-CCCD) are tested with weak classifiers: *k*-NN, support vector machines (SVM) and C4.5 on fake data sets. Prior to the experiment a pilot study is done to get optimal parameters for all the classification



| Data | $q = m/n$ | $N$ | $d$ | | $\sigma = 2$ | $\sigma = 3$ | $\sigma = 4$ | $\sigma = 5$ | $\sigma = 6$ | $\sigma = 7$ | $\sigma = 8$ | $\sigma = 9$ | $\sigma = 10$ |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Sonar | 1.14 | 208 | 61 | OR | 4% | 19% | 23% | 25% | 26% | 26% | 27% | 28% | 28% |
| | | | | IR | 1.22 | 1.04 | 0.96 | 0.93 | 0.96 | 0.97 | 0.97 | 0.90 | 0.90 |
| Ionosphere | 1.78 | 351 | 35 | OR | 25% | 36% | 66% | 69% | 66% | 79% | 61% | 76% | 81% |
| | | | | IR | 90.00 | 62.50 | 8.70 | 6.20 | 5.44 | 3.67 | 5.02 | 3.98 | 3.31 |
| Segment0 | 6.02 | 2308 | 20 | OR | 0% | 0% | 0% | 0% | 0% | 0% | 0% | 0% | 0% |
| | | | | IR | NA | NA | NA | NA | NA | NA | NA | NA | NA |
| Page-Blocks0 | 8.79 | 5472 | 11 | OR | 0.6% | 0.6% | 0.6% | 0.8% | 0.9% | 1% | 1% | 1% | 1% |
| | | | | IR | 0.47 | 0.22 | 0.22 | 0.25 | 0.40 | 0.39 | 0.43 | 0.54 | 0.63 |
| Vowel0 | 9.98 | 988 | 14 | OR | 0% | 0% | 0% | 0% | 0% | 0% | 0% | 0% | 0% |
| | | | | IR | NA | NA | NA | NA | NA | NA | NA | NA | NA |
| Shuttle0vs4 | 13.87 | 1829 | 10 | OR | 0% | 0% | 0% | 0% | 0% | 0% | 0% | 0% | 0% |
| | | | | IR | NA | NA | NA | NA | NA | NA | NA | NA | NA |
| Yeast4 | 28.10 | 1484 | 9 | OR | 45% | 27% | 39% | 37% | 26% | 26% | 26% | 26% | 31% |
| | | | | IR | 18.97 | 99.75 | 18.50 | 18.24 | 392.00 | 390.00 | 391.00 | 390.00 | 41.18 |
| Yeast1289vs7 | 30.70 | 947 | 9 | OR | 45% | 44% | 69% | 43% | 30% | 29% | 29% | 29% | 29% |
| | | | | IR | 24.47 | 23.76 | 24.34 | 23.00 | 47.33 | 45.83 | 45.83 | 45.66 | 45.50 |
| Yeast5 | 32.70 | 1484 | 9 | OR | 6% | 3% | 0% | 0% | 0% | 0% | 6% | 7% | 0.1% |
| | | | | IR | NA | 1.15 | NA | NA | NA | NA | NA | NA | NA |
| Yeast6 | 41.40 | 1484 | 9 | OR | 30% | 46% | 42% | 31% | 30% | 30% | 10% | 13% | 13% |
| | | | | IR | 64.14 | 21.03 | 27.59 | 76.33 | 73.66 | 73.66 | 38.25 | 63.00 | 63.00 |
| Abalone19 | 129.40 | 4174 | 9 | OR | 25% | 20% | 15% | 14% | 13% | 12% | 12% | 11% | 11% |
| | | | | IR | 104.30 | 104.30 | 163.75 | 197.33 | 278.00 | 262.50 | 253.00 | 244.00 | 234.00 |

Fig. 8. *"Overlapping ratios and (local) imbalance ratios in the overlapping region of data sets. " stands for the imbalance ratio in the overlapping region and " stands for the overlapping ratio which is the percentage of points from both classes residing in the overlapping region. IR=" indicates that one of the classes has no members in the intersections of SVDD regions of classes."*, taken from [1].

methods (with exception of C4.5). In the pilot study a Monte Carlo simulation with 200 replications counts how many times $\tau$ value has maximum values with respect to the Area Under the Curve. The best performing $\tau$ is used as the parameter in the main simulation.

The main simulation tests the data sets until AUCs of all classifiers obtain an error below 0.0005. The Average AUC is shown in Figure 7. Observing the classifiers AUCs, the CCCDs perform better than the $k$-NN classifier. With increasing dimensionality the difference becomes more clear, because the class cover gets more complicated. As expected when the class size increases the difference get smaller. This has to do with the fact that the data imbalance gets shifted and the area of both classes are represented in larger amounts [1].

#### 3.1.2 Strong Classifiers

The CCCDs are only compared with strong classifiers on a single simulation settings because CCCD classifiers are observed to be better than other classifiers (C4.5, $k$-NN, SVM). The UAC average results are given in Figure 6. The classifiers are now strong because they are modified in the following way: They now use resampling, ensemble and cost-sensitive schemes. SMOTE+ENN is used as the resampling scheme, EasyEnsmble as the ensembling scheme the algorithms are changed to perceive class weights.

In Figure 6 the strong classifiers are shown together with the CCCDs. RW-CCCDs perform better than the pure CCCDs. Comparing dimensionality with the CCCDs it's visible that in D=10 random walk performs the same or slightly less compared to the EC and C5 ensemble. The random walk CCCDs get results that are comparable to the strong robust methods when there is class imbalance and class overlapping. RW-CCCDs generate prototype sets that reduce the training data set.

### 3.2 Real world

To test the difference between the AUC classifiers, we can compare performance of CCCD classifiers and all other weak and strong classifiers. We test using 5x2 cross validation (CV) which has been found to be the most powerful test among the acceptable type-I error by Dietterich (1998). The test suggest if two classifiers are different in terms of performance.

The overlapping ratios and imbalance in the areas is shown in Figure 8. The test is taken with different data sets for $\sigma = 2, 3, ..., 10$. Some data sets like Ionosphere, Abalone19, Yeast4, Yeast6 and Yeast1289vs7 have more overlap than others and are also more imbalanced. Others do not have this overlap and imbalance.

In real data sets methods to estimate the support and estimate the overlapping ratio is needed because we need to choose the supports of the classes. The method support vector data description (SVDD) finds a region of a data set, that covers a specific percentage of points. The overlapping ratio that shows the percentage of points from both classes that lie in A. By finding the region and overlapping region we can decide where a new point belongs to a particular class.

## 4 Summary and Discussion

In this paper multiple classification methods like: RW-CCCD, pure-CCCD, $k$-NN, SVM and C4.5 have been introduced. Imbalanced data is a recurring problem in real life two-class classification. In other words, a majority class overshadows the minority class, causing biased data. Class imbalances decreases the accuracy of many algorithms also known as Class Cover Problems (CCP). This paper reviewed primarily classification of imbalanced data sets using Class cover catch digraphs (CCCD) using a two-class setting equal to the DeVinny [4] (2003). It turned out that CCCD gave better results when compared to a $k$-NN example, due to the difference in class size, which DeVinney [4] (2003) also wrote in his article. Another method that was tested is P-CCCD classifier where the value $\tau$ has a big influence on the imbalanced data set. Lower values of $\tau$ performed better than a higher value of $\tau$. The last function of CCCD that was tested is RW-CCCD, where a low value of $e$ and a low dimensionality increased the performance a lot.

When looking at the results of CCCD, one may conclude that CCCD is robust to class imbalance. This is demonstrated in the paper by showing different effects of overlapping together with the class imbalance problem. Overlapping affects the performance of imbalanced data sets specifically, when looking at the performance of $k$-NN, SVM and C4.5 are heavily affected by overlapping while CCCD is not severely affected in comparison to these methods. SVM does however perform decently on small imbalanced data sets, but not on highly imbalanced data sets. No matter the imbalance of a data set, CCCD classifiers seem to preserve their AUC compared to $k$-NN, SVM and C4.5 classifiers.

For strong classifiers this paper used SMOTE+ENN which is the most successful method from the families of schemes. The SMOTE+ENN was primarily used for resampling.

## 5 Future research

Imbalanced date is still a huge problem in different areas and will most likely still exist for a long time. This paper gave some examples of methods that solve the issue of class imbalance. However most of these methods are still not fully capable of solving the imbalance issue. Improving the accuracy of classifying algorithms that are able to cope with class imbalance is something to search for. There might even be a scenario where no perfect solution will be found for class imbalance.

## Acknowledgements

The authors wish to thank the main source for this paper *Classification of Imbalanced Data with a Geometric Digraph Family, 2016* [1], and their significant other which provides strength in dark times.

Do not go gentle into that good night,
Old age should burn and rave at close of day;
Rage, rage against the dying of the light.
*Dylan Thomas, 1914 - 1953*

## References

[1] E. C. Artur Manukyan. Classification of imbalanced data with a geometric digraph family. 2016.

[2] T. Cover and P. Hart. Nearest neighbor pattern classification. IEEE transactions on information theory. 1967.

[3] E. Fix and J. L. Hodges. Discriminatory analysis. nonparametric discrimination: Consistency properties. 1989.

[4] D. M. J. DeVinney, C. Priebe and D. Socolinsky. Random walks and catch digraphs in classification. 2002.

[5] J. M. Z. J. Y. L. J. A. B. M. A. Mazurowski, P. A. Habas and G. D. Tourassi. Training neural network classifiers for medical decision making: The effects of imbalanced datasets on classification performance. 2008.

[6] L. O. H. N. V. Chawla, K. W. Bowyer and W. P. Kegelmeyer. Smote: synthetic minority over-sampling technique. 2002.

[7] S. G. V. P. V. Lpez, A. Fernndez and F. Herrera. n insight into classification with imbalanced data: Empirical results and current trends on using data intrinsic characteristics. 2013.

[8] N. V. C. Y. Tang, Y.-Q. Zhang and S. Krasser. SVMs modeling for highly imbalanced classification. 2009.

# An Overview of Visual Place Recognition

Yu-Ying Chen and Zhuoyun Kan

**Abstract**— Visual place recognition aims to distinguish a scene image and deciding whether the place has been visited or not. It has also been a growing research field and crucial in Simultaneous Localization And Mapping (SLAM).
This paper presents an overview on visual place recognition with three techniques, From Accelerated Segment Test and the Binary Robust Independent Elementary Features (FAST+BRIEF), Sequence Matching Across Route Traversals (SMART), and Convolutional Neural Network (CNN) with a generalized layer of Vector of Locally Aggregated Descriptors (NetVLAD). For these techniques, the motivation, the algorithms, the tested datasets, and the performance are discussed.
In brief, NetVLAD performed the best, followed by SMART and FAST+BRIEF. However, a higher computational cost was required to achieve better performance. Future studies of experimenting these techniques with the same dataset are also recommended to gain more insights into the roles of the techniques in visual place recognition.

**Index Terms**—Visual place recognition, FAST+BRIEF, SMART, NetVLAD, SLAM.

## 1 Introduction

Visual place recognition aims to distinguish a scene image and deciding whether the place has been visited or not. In robotics, visual place recognition is highly essential to long-term mobile robot autonomy. It has also been a growing research field and frequently discussed in academic conferences [18, 23]. Also, simultaneous localization and mapping (SLAM) has become increasingly important and been explored for better solutions [6].

However, a number of ongoing challenges of a visual place recognition task have not yet been solved [12]. For example, the appearance of a place can change significantly under different conditions (e.g., different periods of time, illuminations) as shown in Fig. 1(a), while some different places may look similar as demonstrated in Fig. 1(b). Besides, places can be revisited from other positions or distances, which also increases the difficulty of solving a recognition problem.

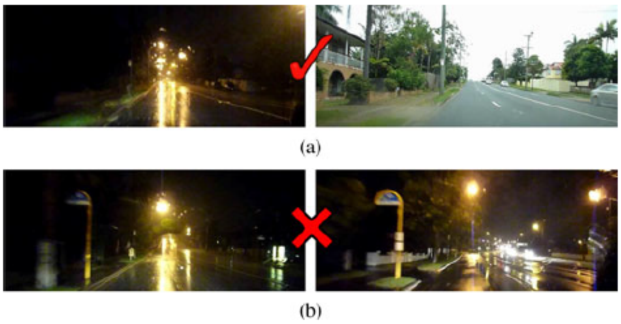


Fig. 1. Challenges of a visual place recognition task: (a) matching images correctly under different conditions and (b) rejecting similar images of different places [12]

The aim of the study was to explore how the three techniques (FAST+BRIEF, SMART, and NetVLAD) performed in a visual place recognition system by their capability of matching perceptually different images while rejecting incorrect matches between aliased images of different places.

In this paper, the theoretical background of visual place recognition is described in Section 2. Section 3 introduces the studies of FAST+BRIEF, SMART, and NetVLAD, and Section 4 presents the experimental results from these studies. The discussion and the conclusion can be found in Section 5 and 6, respectively.

## 2 Theoretical background

According to the literature review, three main modules (Fig. 2) are commonly used in a visual place recognition system [12]. An *image processing* module is responsible for the incoming visual data. A *map* module stores the representation of the robot's knowledge in the real world. A *belief generation* module is used to make a final decision whether a place has been visited before.

Additionally, motion data can also be used as an input to activate the belief generation module. Once the belief generation process finishes, the results can be sent to the map module and update the information in order to implement an online operation [12].

In the following subsection, image processing module is introduced in detail, since it is the major focus of this review paper. In addition, there is another subsection describing the approach to evaluating a visual place recognition system.

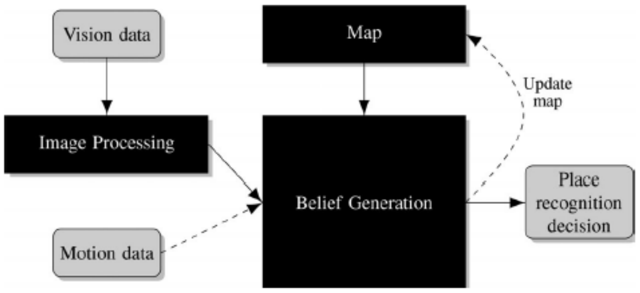


Fig. 2. Schematic of a visual place recognition system [12]

### 2.1 The Image Processing Module

An image processing module is responsible for describing places once the incoming images are detected and stored as vision data in robotic memories. Generally, the techniques of describing features are categorized into two basic types, namely local feature descriptors and global descriptors.

A local feature descriptor is aimed to extract interesting or notable parts from an image. A couple of methods have been proposed to find these local features such as scale-invariant feature transform (SIFT) [11], speeded-up robust features (SURF) [2] and ORB (a very fast binary descriptor based on BRIEF) [19]. Since local feature extraction

- *Yu-Ying Chen is a Master's student in Computing Science at University of Groningen, E-mail: y.chen.66@student.rug.nl.*
- *Zhuoyun Kan is a Master's student in Computing Science at University of Groningen, E-mail: z.kan@student.rug.nl.*

contains two steps, detection followed by description, it is common to combine different techniques (e.g., using FAST detection to find keypoints and BRIEF descriptors to describe keypoints) [12].

Since each image may contain a large number of local features, the bag-of-words model is commonly used to convert the local image features into visual vocabularies by discretizing the feature space [12]. For each image, every feature is represented by a visual word and the spatial structure of the image is ignored. Consequently, this increases the matching efficiency and makes the bag-of-words model work well in pose invariance, which means the robot can recognize the place and make the right decision regardless of its position or orientation. In this case, the robot becomes insensitive to the spacial structure because the geometric space is ignored. However, the implementation of this model increases the risk of losing accuracy in feature matching, especially when the appearance of a place changes drastically. This leads to another challenge in visual place recognition, which is known as condition invariance. This invariance results in robots being reasonably expected to recognize a place successfully when the appearance of the place changes under certain conditions (e.g., illumination). The trade-off between pose invariance and condition invariance is a highly challenging topic in place recognition research [12].

By contrast, a global descriptor focuses on the whole image scene. Global descriptors (e.g., Gist [16]) can be generated by various image features, such as edges or corners, or computed by pre-processed local features with a grid-based pattern. Compared to local feature descriptors, global descriptors perform better in terms of condition invariance. However, they are more sensitive when the robot changes its position and perform worse than local descriptors in this case. A possible way of getting a more accurate result in image processing is combining these two approaches, for example, using bag-of-words model and a Gist descriptor on segments of the whole image [12].

### 2.2 Evaluation of Visual Place Recognition Systems

A visual place recognition system can be evaluated by two measures, precision and recall. Precision is known as the percentage of correct matches among all the image matches (containing both true and false matching results). Recall is defined as the proportion of positive matches that are returned in the system (rejecting some actual matches erroneously).

$$Precision = \frac{TP}{TP+FP} \tag{1}$$

$$Recall = \frac{TP}{TP+FN} \tag{2}$$

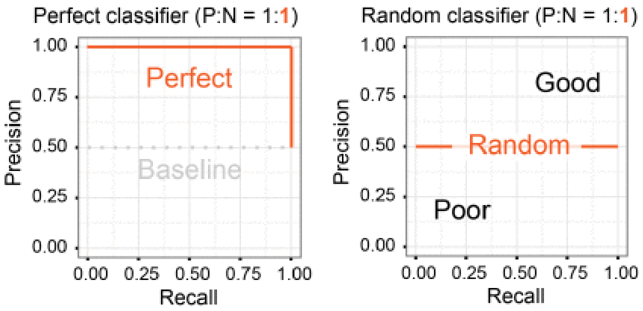
For the above equations, *TP*, *FP*, and *FN* stand for true positives, false positives, and false negatives, respectively. Therefore, the ultimate goal of such a system is to achieve a high value of precision and recall. The optimal result is the realization of 100% of both measures. Precision and recall are typically used together to evaluate the retrieval performance by a precision-recall (PR) curve (Fig. 3).

## 3 Methods

Various methods for recognizing visual places have been proposed in recent years. This section focuses on introducing three techniques, FAST+BRIEF [7], SMART [17], and NetVLAD [1] of solving visual place recognition tasks.

### 3.1 FAST+BRIEF [7]

FAST+BRIEF is a technique using a bag-of-words model in combination with FAST detection and BRIEF description. The study of FAST+BRIEF tried to develop a technique for SLAM, so it was expected to perform real-time visual place recognition and consume less computational power than common algorithms (e.g., SIFT).



(a) An optimal system with 100% precision and 100% recall (b) A poor system with random results

Fig. 3. Precision-recall (PR) curves under different conditions [20]

#### 3.1.1 Algorithm

Since it is highly expensive in terms of computation to extract local features, FAST+BRIEF first processes the images to get FAST keypoints. Then it uses each FAST keypoint to define a square patch, which is used to compute a BRIEF descriptor. The BRIEF descriptor is a binary vector where each bit is the result of an intensity comparison between two of the pixels of the patch. Afterward, the bag-of-words model creates vocabulary tree which is clustered according to the features. Furthermore, the inverse index is kept to store a list of images where each word in the vocabulary is present.

The system takes image query after the previous stage. The bag-of-words vector of the query image would be used to search for the most similar candidates. If the vectors of the query image and the vectors of the previous image reach a given level of difference, this query image would be skipped for possible turning of camera to significantly different directions. The temporal consistency is also checked within a given number of previous queries to make sure the routes have overlaps. Finally, a geometrical check is conducted to find the final candidate with at least 12 correspondences by various approaches (e.g., Flann [15]).

#### 3.1.2 Datasets

NewCollege and Bicocca25b are two datasets used to represent images from different types of environment. NewCollege contains outdoor and dynamic images with a total length of 2260 meters and a revisited length of 1570 meters. Bicocca25b contains indoor and static images with a total length of 760 meters and a revisited length of 113 meters.

#### 3.1.3 Experiments

In the experiments, the ground truth was created manually due to the lack of information regarding loop closures. It provided the correct geometrical information of test images. Besides, FAST+BRIEF, SURF64 [2] (64-D descriptors with rotation invariance), and U-SURF128 [2] (128-D descriptor without rotation invariance) were applied to the training phase, and evaluated by their precision and recall.

### 3.2 SMART [17]

SMART is a sequence-based place recognition algorithm suitable for matching spatially continuous information of images, such as vehicle traveling. Another technique named Sequence Simultaneous Localization And Mapping (SeqSLAM), has been proven to perform place recognition in high accuracy and be feasible in long-distance tasks [14, 22]. However, it has to be performed within some constraints, such as consistent speed of moving. Though there have been other techniques with odometry capable of handling various speeds, they may still fail to recognize the routes with self-similarity (e.g., highways) [5]. Hence, SMART was proposed to deal with both issues.

#### 3.2.1 Algorithm

In the algorithm, the sky blackening is conducted first to remove the sky regions from the images because the sky offers no localization information. The technique of image comparison is then applied to find the overlapping with minimized Sum of Absolute Differences (SAD) score [13], which helps the recognition system deal with the changes of the camera pose. The resolution-reduced and patch-normalized images are used to calculate difference vector D for later verification [14].

In order to reduce the influences from various speeds, SMART learns and queries images at constant distance intervals rather than time intervals. After searching for the image sequences, difference vectors (D) are joined to form an image-matching matrix and constrains the search space to approximately 45° straight lines as shown in Fig. 4. A smaller range of angle takes up less computation time, and with the help of odometry, the line could fit the lowest cost better versus the one without odometry as demonstrated in Fig. 4.

### 3.2.2 Datasets

Two different datasets were used in the experiments, namely Alderley off-road and Surfers Paradise road datasets. The off-road dataset was created by image sequences of routes in woods. Due to the wooded environment found in the Alderly dataset, the data collection process was more challenging compared to the Surfers Paradise dataset. As shown in Fig. 5, one traversal of this dataset was created with various speeds of traveling while the other was created using stable speeds. By contrast, the road dataset was created by image sequences of routes inside the city. Therefore, it provides more distinct contents and objects. This dataset contains one traversal with day scenes and the other with night scenes. Both traversals of road dataset contain a certain level of speed variation due to the peak traffic.

### 3.2.3 Experiments

In the experiments, the ground truth was constructed by manually found pairs of frame correspondences and there were approximately 200 pairs of each dataset with linear interpolation for in-between frames. The off-road dataset was used to investigate the differences between SeqSLAM [14] and SMART, and was used to test the performance of SMART with or without odometry. Besides, the experiments were performed under three scenarios. The first scenario used SeqSLAM, and the second one used Blackening + 2,1 Shifting. The third one used Blackening + 2,1 Shifting with the extra support of odometry. 2,1 Shifting means that the image offset matching was conducted with maximal 2 pixels horizontally and 1 pixel vertically. In addition, the road dataset was used to investigate the effect of changing the sequence length from 5 to 200 meters and find the optimal sequence length in this context.

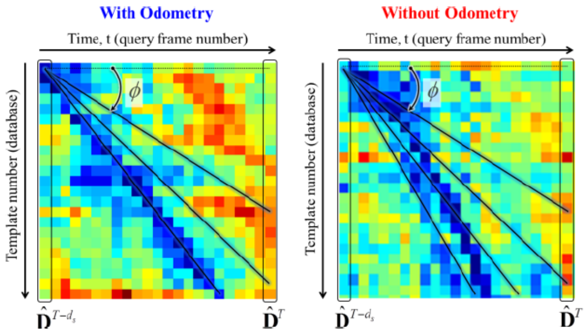


Fig. 4. Cost map showing two different conditions of odometry, the dark blue represents the lowest difference [17]

## 3.3 NetVLAD

The study of NetVLAD introduced a convolutional neural network (CNN) architecture with a trainable generalized VLAD layer, NetVLAD, which was aimed to recognize a given image accurately and quickly [1]. Before the implementation of this architecture, none of the works actually performed the training phase on CNN directly, but treated CNN as black-box descriptor extractors [12]. The study of NetVLAD developed an efficient and flexible learning procedure using hard negative mining and caching [1, 4] and query clustering [1] which can be used to perform large-scale weakly supervised place recognition tasks.

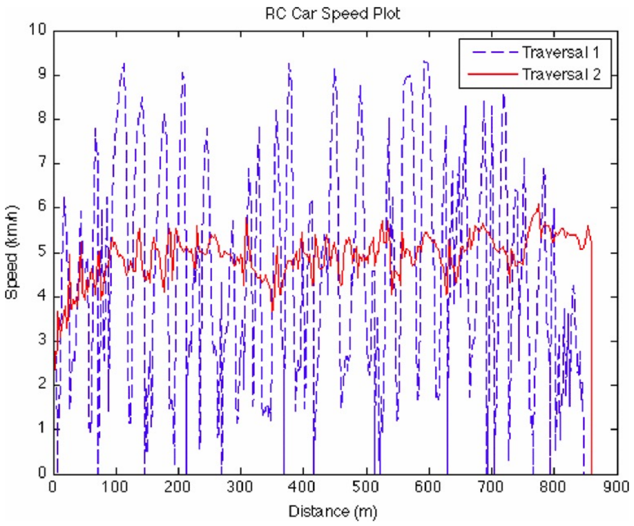


Fig. 5. Various speeds of two traversals of off-road dataset [17]

### 3.3.1 Algorithm

The commonly used image retrieval pipelines basically contain two steps: first extracting local feature descriptors and then pooling them disorderly. According to the standard retrieval pipelines, the entire CNN architecture is designed to be able to learn the image representations end-to-end. The architecture of the NetVLAD layer is shown in Fig. 6 and the input of the NetVLAD layer is the output of the last convolutional layer, which is a set of multidimensional local image descriptors.

VLAD [9] is a descriptor pooling method which extracts information about local descriptors aggregated over the image. In order to obtain a trainable generalized VLAD layer in a CNN framework, NetVLAD is designed and required to be trained via backpropagation, which means this layer's operation is differentiable. However, in the original VLAD layer, the hard assignment of descriptors to the cluster center results in discontinuities [1]. Therefore, a soft assignment of descriptors to multiple clusters is used in order to make the VLAD pooling differentiable. It assigns the weight of the descriptor to the cluster depending on the proximity, and also related to proximities to other cluster centers [1]. After getting VLAD image representations by performing aggregation, which is shown in Fig. 6 as the VLAD core module, the outcome matrix is reshaped by intra-normalization and L2-normalization steps.

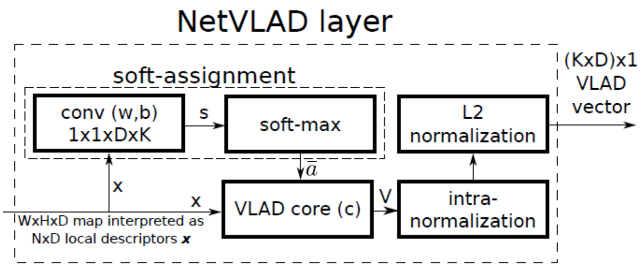


Fig. 6. Architecture of the NetVLAD layer [1]

### 3.3.2 Datasets

NetVLAD was implemented by using Google Street Time Machine as the dataset provider in order to get sufficient amount of data for the training phase. Therefore, a large set of panoramic images were provided, which contained the images of the same locations at different periods of time (e.g., seasons, daytime/night).

Two public datasets were mainly used for the experiments, which were obtained from Pittsburgh (Pits250k) and Tokyo (Tokyo 24/7). The datasets were weakly labeled as they only provided incomplete and noisy supervision. In detail, the Time Machine panoramas only have approximate positions based on (noisy) GPS without information about which parts of the panoramas describe the same scene [1]. Therefore, it makes the visual place recognition a weakly supervised task.

### 3.3.3 Experiments

Experiments were performed on both datasets. For Pits250k, the entire dataset which contained 250k database images was equally segmented for three phases, the training phase, validation phase, and testing phase respectively.

The other dataset, Tokyo 24/7 which contained 76k database images was used as the test set in the experiments. The training set and the validation set were collected additionally to make sure that both of them contained approximately 50k database images. Besides, the images of the test set were taken from three different periods of time (daytime, sunset and night), resulting in a difficult visual place recognition task for the robot.

In addition, Max pooling and NetVLAD layer were used during the experiments. The entire networks were trained by tuning different parameters and layers in order to get optimal outcomes. Besides, three base networks were used to compare with each other: AlexNet [10], VGG-16 [21], and Places205 [24]. The first two were pretrained for ImageNet classification [3]. Places205 used the same architecture as AlexNet, but was pretrained for scene classification [24].

## 4 Results

This section describes different results based on the implementation of three introduced techniques (FAST+BRIEF, SMART, and NetVLAD) and gives the interpretation. These techniques were developed to cope with different scenarios such as the datasets from indoor to outdoor, static to dynamic, day to night, and similar to distinct.

### 4.1 FAST+BRIEF [7]

FAST+BRIEF proposed a way of extracting image features with higher efficiency than SURF. Table 1 gives a summary of the required computational power to obtain geometrical correspondences of different approaches by approximating nearest neighbors in NewCollege dataset. The execution time represents the average time of finishing the geometrical check per query. The N of $DI_N$ stands for the level in the vocabulary tree at which the ancestor nodes are checked. In addition to Flann [15], Exhaustive approach was also used in the experiments. It stands for the exhaustive search that measures the distance of each pairwise features. Though, the exhaustive approach achieved the best recall, $DI_2$ was used as the matching method in the rest experiments as a balanced choice in terms of recall and execution time.

Fig. 7 shows the precision and recall curves of using BRIEF, SURF, and U-SURF descriptors on two datasets. For NewCollege dataset, which contained images of outdoor and dynamic environments, the BRIEF performed a lower recall than either SURF or U-SURF for the same precision value. The inferior outcomes may result from the use of features lacking pose and scale invariance [17]. However, for Bicocca25b dataset, which contained images of indoor and static environments, FAST+BRIEF performed a quite similar recall to SURF and U-SURF for the same precision value. Its elbow reaches the value of 0.9 for both precision and recall. On average, it required 21.6 ms to finish a query [7].

Table 1. Performance of different approaches to obtain correspondences in NewCollege (precision fixed to 100%) [7]

| Technique | Recall (%) | Execution time (ms / query) | | |
|---|---|---|---|---|
| | | Median | Min | Max |
| $DI_0$ | 38.3 | 0.43 | 0.25 | 16.50 |
| $DI_1$ | 48.5 | 0.70 | 0.44 | 17.14 |
| $DI_2$ | 56.1 | 0.78 | 0.50 | 19.26 |
| $DI_3$ | 57.0 | 0.80 | 0.48 | 19.34 |
| Flann | 53.6 | 14.09 | 13.79 | 25.07 |
| Exhaustive | 61.2 | 14.17 | 13.65 | 24.68 |

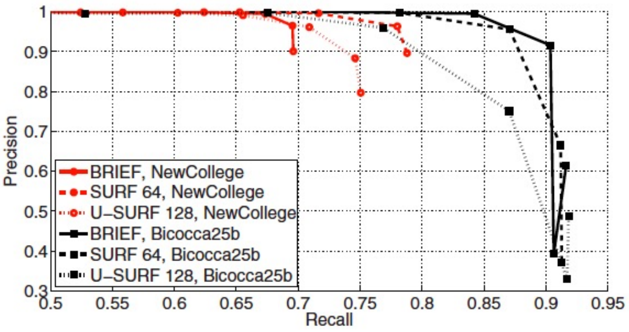


Fig. 7. Precision-recall curve of using BRIEF, SURF, and U-SURF on two datasets [7]

### 4.2 SMART [17]

SMART was developed to have a better performance than SeqSLAM in coping with similar scenes, condition changes, and variation of traveling speeds.

The results of scenario 1 and 2 in Table 2 show that SMART outperformed SeqSLAM in the off-road dataset. The results of scenario 2 and 3 indicate that SMART with odometry could perform better than SMART without odometry in the same dataset. Fig. 8 illustrates the possible different results of place recognition between SMART and SeqSLAM at distinct (top row) and similar (bottom 3 rows) locations. Therefore, it is possible that both SMART and SeqSLAM correctly recognize distinct locations, but only SMART recognizes indistinct images at the same location.

Besides, the road dataset was used to test the effects of changing the length of the sequence. Table 3 shows the performance of different sequence lengths varying from 5 to 200 meters. The performance increased significantly when the length reached 30 meters, and the highest recall of 96% occurred when the length was 100. However, the performance dropped dramatically at a sequence length of 200 because of the accumulated odometry error with a long sequence length (e.g., 200 meters). Hence, incorrect matches occurred when fitting straight lines across slightly non-linear sequences. It is also noticeable that the maximum possible recall dropped from 100% to 91% when the sequence length increased.

Table 2. Off-Road Scenario testing results [17]

| No. | Scenario | Best Recall at 100% Precision |
|---|---|---|
| 1 | SeqSLAM | 1% |
| 2 | SMART: Blackening + 2,1 Shifting | 25% |
| 3 | SMART: Odo + Blackening + 2,1 Shifting | 36% |

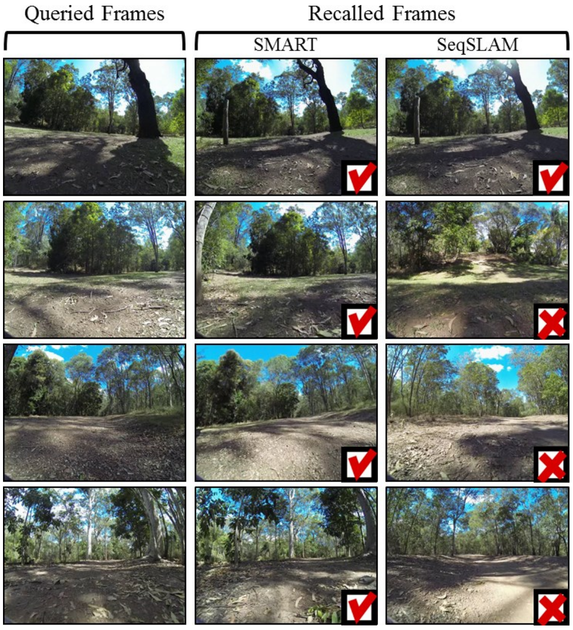


Fig. 8. Examples of similar images recognized by SMART and SeqSLAM [17]

Table 3. Sequence length performance [17]

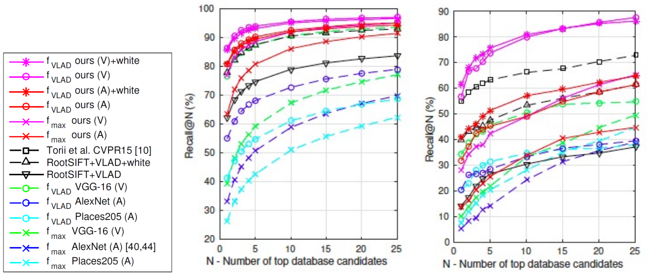
| Sequence Length (1 frame/meter) | Best Recall Achieved at 100% Precision | Maximum Possible Recall at 100% Precision |
|---|---|---|
| 5 | 12% | 100% |
| 10 | 37% | 100% |
| 20 | 47% | 99% |
| 30 | 81% | 99% |
| 40 | 86% | 98% |
| 50 | 88% | 98% |
| 75 | 91% | 97% |
| 100 | 96% | 96% |
| 200 | 68% | 91% |

### 4.3 NetVLAD

NetVLAD is an architecture of CNN for weakly supervised tasks and could provide quick and accurate visual place recognition results with good pose and condition invariance.

The comparison results can be seen in Fig. 9, depicting the performances of different methods mainly on two datasets (Pits250k and Tokyo 24/7). It is obvious that the trained representations (red and magenta lines for AlexNet and VGG-16) performed better than off-the-shelf ones (blue, cyan, green lines for AlexNet, Places205, VGG-16). Besides, the implementation of NetVLAD pooling ($f_{VLAD}$) outperformed the Max pooling ($f_{max}$) for both trained and off-the-shelf representations [12]. The study also indicated that training the NetVLAD layer could contribute to a great improvement in the final recognition results, but training other layers may take the risk of overfitting and influence the performance [1].

When comparing Pits250k and Tokyo 24/7, it is clear that the overall results vary a lot between the two datasets. A better accuracy was achieved For Pits250k. This may result from the different degrees of complexity of the database images. Tokyo 24/7 contains query images taken during sunset and night, which increases the difficulty in condition invariance due to the significant difference of illumination. Therefore, it is easy to have incorrect matches between aliased images which look similar to each other in different environments.



(a) Marks of different methods (b) Pittsburgh (Pits250k) (c) Tokyo 24/7

Fig. 9. Comparison of different methods (NetVLAD, off-the-shelf networks and state-of-the-art) [1]

## 5 Discussion

The similarities and differences of FAST+BRIEF [7], SMART [17], and NetVLAD [1] are given in this section. Besides, the strengths and weaknesses of these techniques under different conditions are discussed.

### 5.1 Similarities and Differences in Various Aspects

All of these techniques cast visual place recognition as image retrieval tasks and focus on the place re-localization within a map. They work on extracting features based on different techniques to improve performance in terms of pose and condition invariance. As can be seen from the results mentioned above, all these studies demonstrate a certain degree of reliability and efficiency of place recognition tasks.

However, the focused fields of these techniques differ from each other. FAST+BRIEF is aimed to quickly extract the BRIEF descriptors and conduct geometrical verification by bag-of-words model. SMART focuses on image sequences, especially for traveling vehicles in the dynamic environment with various moving speeds. NetVLAD is a kind of learning-based approach and makes great improvements on traditional learning tasks. It is designed to enable end-to-end learning of the network architecture, which trains the image representations directly by tuning parameters for the end-task.

### 5.2 Strengths and Weaknesses

FAST+BRIEF was able to perform similarly to SURF with faster execution time in the indoor static environment. Another advantage was its ability to be highly accurate in the same environment while keeping most executions with much less time than using exhaustive approach. Therefore, FAST+BRIEF could help SLAM systems detect closing loops with higher efficiency without requiring any specialized hardware. However, the results in the outdoor dynamic environment also exposed the weakness of FAST+BRIEF, which is the limited ability to achieve pose and condition invariance. Under these circumstances, having the support from other techniques or state-of-the-art methods would be essential for improving performance.

SMART, an improved technique of SeqSLAM [14], could perform well in outdoor environments with moderate pose and condition invariance. The blackening process was used to better handle the change of lighting conditions for both day and night scenes. Different from SeqSLAM, it was able to handle the speed variation with the use of odometry. When the sequence length was higher than 30 meters, its recall reached more than 80% for the road dataset. It also achieved nice performance in an outdoor environment with similar scenes. The recall of SeqSLAM was only 1%, but the recall of SMART reached 25% with only camera images and 36% with additional odometry. Even though the recall was not high enough to be feasible in real-world applications, it established a good start in coping with highly similar environments. Though the SMART could perform well in urban traversals, it requires high computational power to support a real-time navigating system in a large area with complex roads. Hence, the author suggested combining external localization information (e.g., GPS) to restrict the search to a certain range of area.

NetVLAD performed well even in extreme environment which contained day, night, and sunset images [1]. It excelled at identifying regular urban query images, with a recall higher than 80%, and was able to obtain a recall higher than 60% for the extremely challenging dataset (Tokyo 24/7 sunset/night) while the other off-the-shelf descriptors could only have a recall at approximate 30%. In general, NetVLAD is a very powerful technique in place recognition due to its good pose and condition invariance [8]. It can be used for very large scale weakly supervised place recognition tasks. The major drawback of using NetVLAD could be the requirement of large sets of training images for an accurate result, which makes the whole process computationally expensive. The performance of an application with the implementation of NetVLAD would be determined critically by the available datasets. The learning phase in the neural network also takes up a huge amount of time.

## 6 Conclusion

It is clear to see the improvements among these three techniques. Regarding the pose and condition invariance, FAST+BRIEF is sufficient to handle indoor and static scenario, but hard to handle outdoor and dynamic scenario. SMART is capable of recognizing outdoor and dynamic scenario, such as day and night images by its moderate pose and condition invariance. NetVLAD can be applied to large-scale weakly supervised tasks, and it is capable of dealing with images taken during the day, night, and sunset by its higher pose and condition invariance.

In regard to computational complexity, FAST+BRIEF was able to process a query image to find matches within short time [7], and this makes it possible to realize SLAM. For SMART, the complexity would grow higher when the dataset becomes larger, which makes it harder to implement in a SLAM system. Therefore, it is suggested to use external localization system (e.g., GPS) to narrow the search space. For NetVLAD, it requires large training time before being able to take query images, and the training time would increase significantly when the dataset is large or grows fast. In general, a higher computational cost is required to achieve better performance.

Finally, in order to acquire more thorough understanding of the conditions of applying each technique, more empirical studies related to the approaches of each technique should be taken into account, and it would be recommended to evaluate the given conditions in terms of the environment of tasks, the available equipment, and the images for training. Future studies of experimenting these techniques with the same dataset are also recommended to gain more insights into the roles of the techniques in visual place recognition.

## Acknowledgements

The authors wish to thank the expert reviewers, Nicola Strisciuglio and Maria Leyva, and the anonymous fellow reviewers for improving this paper.

## References

[1] R. Arandjelovic, P. Gronat, A. Torii, T. Pajdla, and J. Sivic. Netvlad: Cnn architecture for weakly supervised place recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PP(99):1–1, 2017.

[2] H. Bay, A. Ess, T. Tuytelaars, and L. Goo. Speeded-up robust features (surf). *Computer Vision and Image Understanding*, 110(3):346–359, 2008.

[3] J. Deng, W. Dong, R. Socher, L. J. Li, K. Li, and L. Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 248–255, June 2009.

[4] P. F. Felzenszwalb, R. B. Girshick, D. McAllester, and D. Ramanan. Object detection with discriminatively trained part-based models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(9):1627–1645, Sept 2010.

[5] G. Floros, B. van der Zander, and B. Leibe. Openstreetslam: Global vehicle localization using openstreetmaps. In *2013 IEEE International Conference on Robotics and Automation*, pages 1054–1059, May 2013.

[6] J. Fuentes-Pacheco, J. Ruiz-Ascencio, and J. M. Rendón-Mancha. Visual simultaneous localization and mapping: a survey. *Artificial Intelligence Review*, 43(1):55–81, Jan 2015.

[7] D. Galvez-López and J. D. Tardos. Bags of binary words for fast place recognition in image sequences. *IEEE Transactions on Robotics*, 28(5):1188 – 1197, 2012.

[8] H. Jin Kim, E. Dunn, and J.-M. Frahm. Learned contextual feature reweighting for image geo-localization. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017.

[9] H. Jgou, M. Douze, C. Schmid, and P. Prez. Aggregating local descriptors into a compact image representation. In *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 3304–3311, June 2010.

[10] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 25*, pages 1097–1105. Curran Associates, Inc., 2012.

[11] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60(2):91–110, Nov 2004.

[12] S. Lowry, N. Sunderhauf, P. Newman, J. J. Leonard, P. C. D. Cox, and M. J. Milford. Visual place recognition: A survey. *IEEE Transactions on Robotics*, 32(1):1–19, 2016.

[13] M. J. Milford, F. Schill, P. Corke, R. Mahony, and G. Wyeth. Aerial slam with a single camera using visual expectation. In *2011 IEEE International Conference on Robotics and Automation*, pages 2506–2512, May 2011.

[14] M. J. Milford and G. F. Wyeth. Seqslam: Visual route-based navigation for sunny summer days and stormy winter nights. In *2012 IEEE International Conference on Robotics and Automation*, pages 1643–1649, May 2012.

[15] M. Muja and D. G. Lowe. Fast approximate nearest neighbors with automatic algorithm configuration. In *In VISAPP International Conference on Computer Vision Theory and Applications*, pages 331–340, 2009.

[16] A. Oliva and A. Torralba. Modeling the shape of the scene: A holistic representation of the spatial envelope. *International Journal of Computer Vision*, 42(3):145–175, May 2001.

[17] E. Pepperell, P. I. Corke, and M. J. Milford. All-environment visual place recognition with smart. In *Robotics and Automation (ICRA)*. IEEE, 2014.

[18] E. Pepperell, P. I. Corke, and M. J. Milford. Automatic image scaling for place recognition in changing environments. In *2015 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1118–1124, May 2015.

[19] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski. Orb: An efficient alternative to sift or surf. In *Computer Vision (ICCV)*. IEEE, 2011.

[20] T. Saito and M. Rehmsmeier. The precision-recall plot is more informative than the roc plot when evaluating binary classifiers on imbalanced datasets. *PLOS ONE*, 10(3):1–21, 03 2015.

[21] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *CoRR*, abs/1409.1556, 2014.

[22] N. Sünderhauf, P. Neubert, and P. Protzel. Are we there yet? challenging seqslam on a 3000 km journey across all four seasons. In *IEEE International Conference on Robotics and Automation (ICRA) 2013*, Proc. of Workshop on Long-Term Autonomy, IEEE International Conference on Robotics and Automation (ICRA), Karlsruhe, Germany, 2013.

[23] A. Torii, R. Arandjelovi, J. Sivic, M. Okutomi, and T. Pajdla. 24/7 place recognition by view synthesis. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1808–1817, June 2015.

[24] B. Zhou, A. Lapedriza, J. Xiao, A. Torralba, and A. Oliva. Learning deep features for scene recognition using places database. In *Advances in Neural Information Processing Systems 27*, pages 487–495. Curran Associates, Inc., 2014.

# Simultaneous Localization and Mapping (SLAM) for Robots: An Introduction and Comparison of Methods

Shaniya Hassan Ali and Hatim Alsayahani

**Abstract**—Mapping the structure of an unknown environment by a robot and navigating within this generated map is commonly called SLAM. Earlier work on SLAM made use of a laser rangefinder and wheel odometer which came with downsides such as expensive equipment and bad readings from certain surfaces. However, in recent years visual SLAM, where the primary mode of sensing is via a camera, have become the more dominant choice and have been the object of active research over the past two decades.

In this paper, we will look into the different methods for visual SLAM implementation from *Stereo Parallel Tracking and Mapping* to *ORB-SLAM*. This will give a good overview of the current approaches to SLAM. The paper will also present any other alternate visual methods that can be used which have been overlooked.

The main research done on these approaches will be studied, as well as discussed, highlighting the pros and cons of each implementation. We will also examine the tests done to prove the effectiveness of these methods. Furthermore, importance will be given to the performance and complexity constraints which will draw special attention to the geometric limitations such as pure rotation in monocular, baseline in stereo slam, camera hardware limitations such as limited bandwidth and/or frame-rate, and misreading objects under varying conditions.

**Index Terms**—Slam, visual, stereo, lsd-slam, orb-slam, image parsing, cameras, navigation.

## 1 Introduction

Simultaneous localization and mapping from now on referred to as SLAM is not a particular algorithm or piece of software, but rather a concept. It is commonly compared to the chicken-or-egg conundrum as it refers to the problem of trying to concurrently estimate the orientation of some sensor with respect to its surroundings, while at the same time mapping the structure of that environment. In practice, these two problems cannot be solved independently of each other [10]. Discerning where a robot is within its environment and figuring out what that environment looks like is hard to estimate without a map. A good map is needed for localization while an accurate pose estimate is needed to build that map.

Much of the early research on SLAM involved ground based robots equipped with laser scanners. This illustrates many of the core difficulties of SLAM, such as creating a consistent and accurate map, and making best use of multiple unreliable sources of information. It also came with disadvantages such as expensive equipment and certain surfaces which can give bad readings. More recently the use of visual sensors have become the more dominant choice in SLAM research, partly because an image provides a rich source of information about the structure of the environment. In the recent decades, a large amount of research on visual SLAM used stereo cameras, or cameras alongside other sensors (such as accelerometers or GPS), but since around 2001 a number of works have shown how SLAM could be done successfully using only a single camera (known as monocular visual SLAM).

This was crucial in making SLAM a more widely useful technology, since devices equipped with just a single camera - such as web-cams and mobile phones - are vastly more common and accessible than specialized sensing equipment. Also cameras are passive sensors and do not interfere with each other when deployed in the same environment. In addition, cameras can be used in both indoor and outdoor environments. SLAM is an active field of research within computer vision and new and improved techniques are constantly emerging.

## 2 Challenges

Constructing a practical and stable SLAM system can be very challenging. In this paper, our focus is on modern visual SLAM methods and hence problems related to monocular, stereo and camera hardware limitations will be briefly over-viewed in the coming sections. They are not restricted to specific SLAM methods as these are universal to visual SLAM where camera is the main source of input. The different limitations introduced by hardware, geometric considerations, device or scene motion and so on, suggests the need for specific SLAM systems tailored towards different applications and platforms, such as indoor/outdoor, mobile phone/headset/automobile. No one general SLAM system can deal with all the presented situations.

### 2.1 Bandwidth

In reference to camera frames, there is typically a very limited bandwidth to send and receive them. Hence we can either use relatively low-resolution image data, or compress frames. However, both options are detrimental to the performance of SLAM systems. One such adverse affect is the difficulty to extract minute details within the scene from low-resolution images.

### 2.2 Frame-rate

Due to movement of the cameras in SLAM systems, which is key to navigation within a map, there is a considerable amount of blurriness introduced into the frames captured by the cameras. Although the motion blur can be reduced by increasing the frame-rate from the standard frame-rate of 30 frames per second, this will ultimately effect the bandwidth and computational power.

### 2.3 Pure Rotations in Monocular SLAM

Pure rotations in monocular SLAM can lead to problems in camera pose. It doesn't provide enough information to triangulate new map points, and scenes can change very quickly, faster than translations. To triangulate a point, we need to observe it from two different places. This can be reduced by rotating to new areas very slowly and making concurrent moves. Pure rotation is specifically bad only for feature-based non-filter based visual monocular SLAM or key frame based SLAM.

- *Shaniya Hassan Ali is a Computing Science Master student at the University of Groningen, E-mail: S.Hassan.Ali@student.rug.nl.*
- *Hatim Alsayahani is an academic research at king abdulaziz city for science and technology and a Computing Science Master student at the University of Groningen, E-mail: H.B.Alsayahani@student.rug.nl.*

### 2.4 Baseline in Stereo SLAM

Baseline is the spatial distance between the two cameras in the system. Stereo SLAM systems can have at least two cameras, hence a good estimation of scene depth needs a decent baseline. Generally, the wider the baseline, the better the depth estimation. However, having a system with a wider baseline would require a larger space to setup the cameras, which might not always be available.

## 3 Image Representation

Visual SLAM algorithms are designed to take advantage of the very rich information about the world available from image data. The way that SLAM systems use these data can be classified as sparse/dense and direct/indirect [10]. The former describes the quantity of regions used in each received image frame, and the latter describes different ways in which the image data are used. This means there are four possible types of SLAM system, in terms of how the image data are used.

### 3.1 Sparse and Dense

Sparse is used by selecting a small subset of the pixels from an image frame. It basically reduces the image to a set of sparse key points that can be used to match with feature descriptors. It is most useful in tracking the camera pose as maps generated from sparse images are point clouds as shown in Figure 1. The advantages of this method are image to geometry transition which is easier while having a broad baseline matches. In contrast, certain disadvantages of sparse method is that it only creates a limited map of the world and does not sample across all available image data such as edges or weak intensity points.

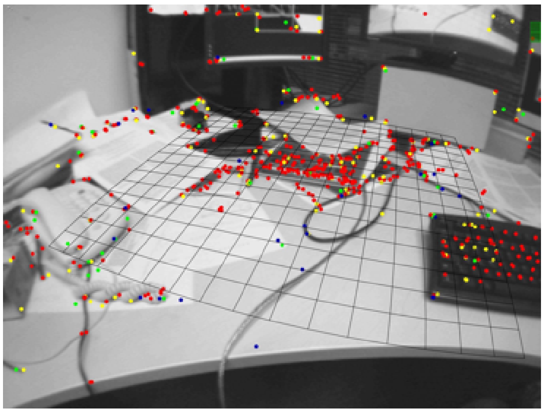


Fig. 1. Sparse map. Colored points on the image are map points.

The dense method uses most or all of the pixels from an image frame. Due to this factor, it requires more powerful hardware such as a GPU. However, it is able to provide much more details of the viewed images as shown in Figure 2 which is an added advantage as it minimizes the photo-metric error. While it gives a more meaningful representation to the human eye by providing more information about the environment, dense methods cannot handle outliers very well. Therefore, this method is computationally very intense and hence much more slower than sparse based methods.

### 3.2 Direct and Indirect

How SLAM systems utilizes the information received from the image can be classified as direct or indirect SLAM. The indirect method relies on feature extraction and uses these features to locate the camera and build the map as seen in Figure 3. Examples of features that are extracted corners/edges or by using special feature descriptors such as SIFT, ORB, FAST, etc. One such implementation can be seen in Figure 4. The main advantage of such an approach is its tolerance towards changing lighting conditions. However it is designed to work with few features and not to use as many data as possible to describe the environments

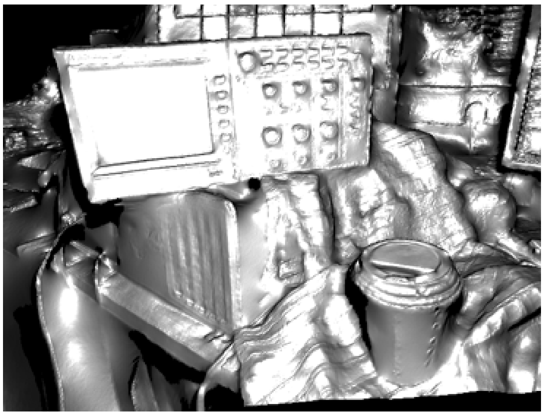


Fig. 2. Dense map. All points on the image are map points.

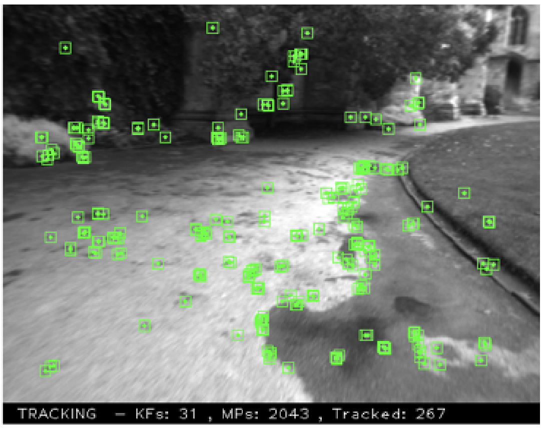


Fig. 3. Indirect Method. Extracting special features from image.

Direct, on the other hand uses pixel intensities directly. What this means is that it can directly compare the intensity between the pixel in one image and its warped projection in another image, which is referred to as photo-metric error [4]. It is able to recover the environment depth, structure and camera pose using this method. The advantage it has over indirect methods is the maintaining the same frame rate allowing more time for other computations.

## 4 Visual Methods

Visual SLAM is a set of SLAM methods that use visual information (e.g. from a monocular camera) to solve the SLAM problem [1]. It is the process of determining the position and orientation of a robot by analyzing the associated camera images. We have selected four methods of visual SLAM which will be briefly described in the next section highlighting the main issues with each and these methods will be compared in the later section.

### 4.1 ORB-SLAM

ORB-SLAM is indirect method and feature-based monocular SLAM system. It runs in real-time for both indoor and outdoor environments.

The method works on three threads, a tracking thread, a local mapping thread and a loop closing thread [6]. Initializing the map is done by computing the relative pose between two sections , computing two geometrical models in parallel, a homography for a planar section, and a fundamental matrix for a non-planar section. Tracking is used to know when to insert a new keyframe and localizes the camera . Features are matched with the previous frame and the pose is optimized using motion-only bundle adjustment.Therefore , the covisibility graph of keyframes is maintained by the system and used to get

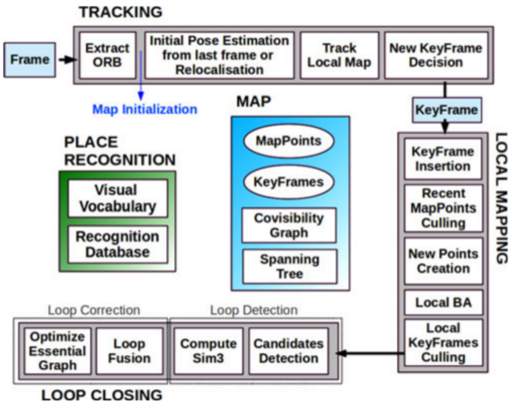


Fig. 4. Overview over the complete ORB-SLAM System.

a local visible map which consists of keyframes to share map points with the current frame.Loop closing is bag of words vectors is used to check possible loops, the smaller bag of words vectors is chosen as benchmark . The current keyframe is removed when all the keyframes with a bag of words similarity to the current key frame that is greater than this benchmark.

To design ORB-SLAM from scratch i.e., a novel monocular SLAM system whose main contributions mainly are as follows.
1-Uses same features for all tasks: tracking, mapping, relocalization and loop closing which helps the system to be our system more efficient, simple, and reliable.
2-Real time operation in large environments.
3-Real time loop closing based on the optimization of a pose graph.
4-Real time camera relocalization with significant invariance to viewpoint and illumination which helps to recover from tracking failure and enhances map reusability.
5-New initialization procedure based on model selection.
6-A survival of the fittest approach to map point and keyframe selection.

### 4.2 S-PTAM SLAM

The main idea of Stereo Parallel Tracking and Mapping (S-PTAM) is to detect the same visual point-landmarks on a pair of synchronized images, which can be used to recover their depth information and incrementally build on a sparse point-cloud representation of the environment. It follows the same approach as Parallel Tracking and Mapping (PTAM) in which the task is separated into two parallel tracks i.e camera tracking and map optimization, sharing only the map between them.

Early work on S-PTAM relied on probabilistic approaches such as Extended Kalman Filter and Particle Filters[7]. Comparison between filter-based and keyframe-based methods which used global optimization techniques like Bundle Adjustment, found that the latter achieves the best balance between precision and computational cost.

In the tracking thread, *feature extraction* is done on the local image as shown in Figure 5. These features are matched to existing map-points and tracked for each frame, used to create new ones by matching stereo correspondences. *Matching* is performed on the descriptors found in a close neighborhood of the points projection on the image. Pose refinement and keyframe selection are the consequent actions taken on it. A keyframe is selected when the number of tracked points is less than 90(%) of the points tracked in the last keyframe. On the parallel thread, map optimization is performed via Bundle Adjustment. It is a simultaneous refinement of a set of camera poses and 3D points reducing the re-projection error of every point in every image.

### 4.3 LSD-SLAM

Large-Scale Direct Monocular SLAM, henceforth referred to as LSD-SLAM, is a type of monocular slam method which performs dense or

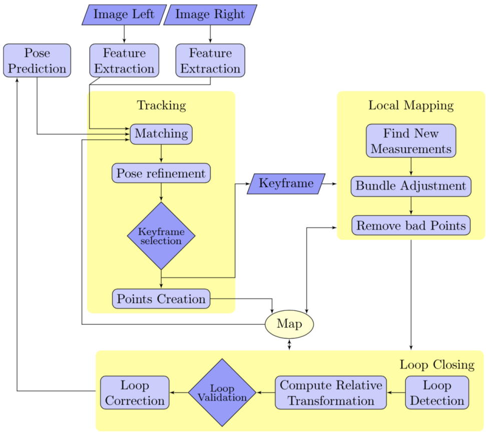


Fig. 5. Overview of S-PTAM process.

semi-dense reconstruction of the scene based on certain key-frames. It uses the intensities of the image to track and map rather than using key-points to create an abstraction of the images being inputted [2]. This allows for mapping a large area and is hence considered to be a powerful method as it doesn't require any specialized hardware. The algorithm has three components which are tracking, depth map estimation and map optimization [2]. An overview of the LSD process is shown in Figure 6.

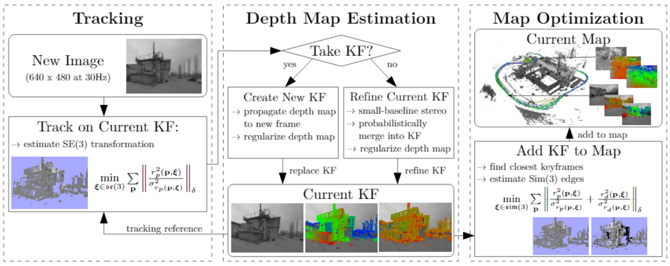

Tracking
New Image
(640 x 480 at 30Hz)
Track on Current KF:
→ estimate SE(3) transformation
$\min_{\xi \in se(3)} \sum_{\mathbf{p}} \left\| \frac{r_p^2(\mathbf{p},\xi)}{\sigma^2_{r_p(\mathbf{p},\xi)}} \right\|_\delta$
tracking reference
Depth Map Estimation
Take KF?
yes
no
Create New KF
→ propagate depth map to new frame
→ regularize depth map
Refine Current KF
→ small-baseline stereo
→ probabilistically merge into KF
→ regularize depth map
replace KF
refine KF
Current KF
Map Optimization
Current Map
add to map
Add KF to Map
→ find closest keyframes
→ estimate Sim(3) edges
$\min_{\xi \in sim(3)} \sum_{\mathbf{p}} \left\| \frac{r_p^2(\mathbf{p},\xi)}{\sigma^2_{r_p(\mathbf{p},\xi)}} + \frac{r_d^2(\mathbf{p},\xi)}{\sigma^2_{r_d(\mathbf{p},\xi)}} \right\|_\delta$

Fig. 6. Overview of LSD-SLAM algorithm.

New camera images are tracked continuously using direct tracking. Their position in the current keyframe is estimated using the previous frame's pose [3].

Depth map estimation uses tracked frames to either refine or replace the current key-frame. a new key-frame is taken based on one condition. If the sum of the distance and the angle to the current key-frame is larger than a certain threshold.

Map optimization tends to improve the map by using pose graph optimization to continuously optimize the background of the map.

A common issue with monocular SLAM is that the scale of the world is not visible which can cause issues when moving from small to large scale and vice versa. LSD SLAM avoids this issue by using the fact that the scene depth and the tracking accuracy correlate to work out the scale of the scene being tracked. At its creation, LSD-SLAM was faster than other monocular SLAM types, but as of 2015 it has been overtaken by feature based methods (using corners). Another disadvantage of LSD-SLAM is that use of direct methods mean that it isn't very flexible, making it harder to remove outliers in the collected data [9].

### 4.4 Seq-SLAM

Mainly a vision based localization that works well under changing conditions such as daytime to nighttime, season to season and from clear weather to rain. The main advantage of sequential SLAM is that it can perceive changes across extreme environmental changes. This is achieved by picking the best match within any short sequence of images.

Seq-SLAM was inspired from early work on Rat-SLAM which also achieved localization over sequences, rather than individual places. The requirement for perfect data association or feature detection is removed by this method. However it lacked versatility and implementation was on a road network mapped during the day which will not be reliable during the night. Seq-SLAM improves on this implementation by catering to the changing conditions, geometry or features.

Localization is accomplished by recognizing coherent sequences of these ”local best matches” in comparison to ”global best matches”. This can be seen in Figure 7 where images captured from different angle and weather conditions are able to correctly match. A template learning was synthesized by capturing one in every 4 original video frames while reducing the learning while stationary. Tests carried out on two varying datasets by Michael J Milford [5] displayed a matching trajectory segments with 100(%) accuracy and precision with recall rates of up to 60(%).

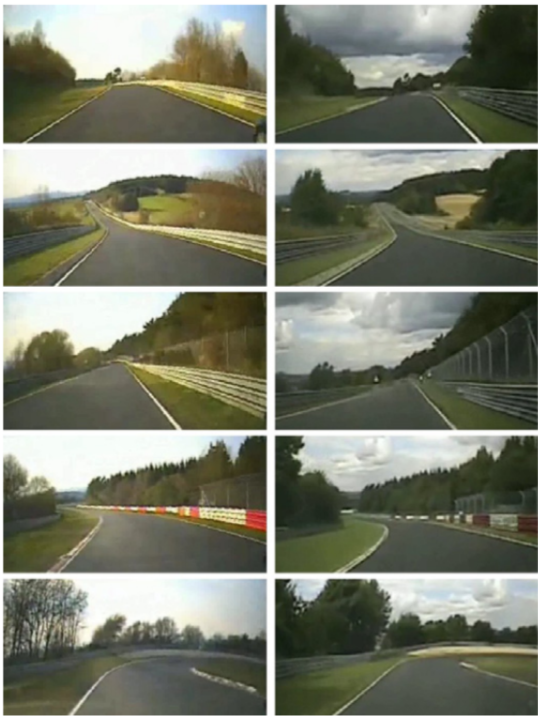


Fig. 7. Corresponding frames from a matched sequence in the Nurburgring dataset.

## 5 Comparison

The four methods described in section 4 will be compared based on performance and complexity of implementation.

### 5.1 Performance

It is hard to compare the methods performance because it have different outputs.Real-time monocular SLAM algorithms such as LSD-SLAM has certain disadvantages because they use the direct method. The photo-metric consistency limits the baseline of the matches. This has a great impact in reconstruction accuracy, which requires wide baseline observations to reduce depth uncertainty. Another disadvantage is that it is, in general, very computationally demanding , which affects the performance. The map optimization is reduced to a pose graph, discarding all sensor measurements. In contrast, feature-based methods such as ORB-SLAM have superior accuracy over direct methods. They are able to match features with a wide baseline, thanks to their good in variance to view-point and illumination changes. BA jointly optimizes camera poses and points over sensor measurements [6].

In this website [11] an experiment done by recorded a sequence at 20 Frames per second (FPS) to compare LSD-SLAM and ORB-SLAM . The observations are LSD-SLAM is semi-dense and not very noisy, which is more useful for 3d-reconstruction. The ORB-SLAM, gives an excellent odometry and is robust to considerable rotation per translation and if tracking is lost it relocalises with little delay as soon as frames have enough overlap with old frames.

While the accuracy of the S-PTAM method was tested in public outdoor and indoor datasets, comparing results against the provided ground truth where it was available [7]

### 5.2 Complexity

The difference between the EKF-based mapping in MonoSLAM and the BA-based mapping with the key-frames in PTAM was discussed in the literature [8]. According to the literature, to improve an accuracy of visual SLAM, it is important to increase the number of feature points in a map. From this point of view, the BA-based approach is better than the EKF-based approach because it can handle large number of points.

## 6 CONCLUSION

This paper has presented types of Image representation.However, our aim has been to present SLAM and identify important visual methods issues to consider. like geometric challenges such as Pure rotations in monocular SLAM which can lead to problems in camera pose and baseline in Stereo SLAM having a system with a wider baseline would require a larger space to setup the cameras, which might not always be available. Camera hardware challenges such as camera frames, there is typically a very limited band-width and Frame-rate. The paper also compare the methods regarding performance and complexity.
Developer must carefully consider the difference between the methods they use and they should consider using visual methods where appropriate.

### Acknowledgements

We would like to thank the expert reviewer, Nicola Strisciuglio, for the valuable insight and suggestions on reviewing this paper. Our esteemed gratitude to also our peer reviewers, Siebert Elhorst and J. Boonstra, who have provided useful feedback.

### References

[1] K. K. Artur Huletski, Dmitriy Kartashov. slam visual methods.

[2] J. Engel, T. Schöps, and D. Cremers. Lsd-slam: Large-scale direct monocular slam. In D. Fleet, T. Pajdla, B. Schiele, and T. Tuytelaars, editors, *Computer Vision – ECCV 2014*, pages 834–849, Cham, 2014. Springer International Publishing.

[3] J. Engel, J. Sturm, and D. Cremers. Semi-dense visual odometry for a monocular camera. In *Proceedings of the 2013 IEEE International Conference on Computer Vision*, ICCV '13, pages 1449–1456, Washington, DC, USA, 2013. IEEE Computer Society.

[4] M. Irani. All about direct methods. 02 2000.

[5] M. J. Milford and G. F. Wyeth. Seqslam: Visual route-based navigation for sunny summer days and stormy winter nights. In *2012 IEEE International Conference on Robotics and Automation*, pages 1643–1649, May 2012.

[6] R. Mur-Artal, J. M. M. Montiel, and J. D. Tards. Orb-slam: A versatile and accurate monocular slam system. *IEEE Transactions on Robotics*, 31(5):1147–1163, Oct 2015.

[7] T. Pire, T. Fischer, J. Civera, P. D. Cristforis, and J. J. Berlles. Stereo parallel tracking and mapping for robot localization. In *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1373–1378, Sept 2015.

[8] H. Strasdat, J. M. M. Montiel, and A. J. Davison. Editors choice article: Visual slam: Why filter? *Image Vision Comput.*, 30(2):65–77, Feb. 2012.

[9] M. T. Tombone. The future of real-time slam and deep learning vs slam.

[10] website OpenSLAM. What is slam.

[11] J. wordpress website. Orb-slam vs lsd-slam.

# An Overview Of The Enterprise Adoption Of Cloud Computing, From Its Early Stages To Now

Kyprianos Isaakidis, Marios Lykiardopoulos

**Abstract**—Cloud Computing since its early stages has been seen as a useful tool for enterprises world-wide. Cloud service providers are offering a multitude of delivery models ( Saas, PaaS, IaaS) and deployment models(public, community, private, hybrid), to enable enterprises to scale their IT products and/or infrastructure. When enterprises decide to either migrate or build their whole or part of their infrastructure to the Cloud, they invest a lot of resources into making the right choice. What affects this decision is influenced by many internal and external factors. In this paper we look into factors from the early stages of Cloud Computing's lifetime, to the recent years, and see how these factors have changed over this period of time. Since adopting Cloud Computing on an enterprise level is a complicated matter, it is interesting to see which of the Cloud Computing adoption factors have remained important throughout the aforementioned period of time. We also investigate what factors are important to enterprises of different sizes. The analysis of these factors, is cross validated from empirical data, from the relevant bibliography.

**Index Terms**—Cloud Computing adoption, SME, large enterprise.

## 1 Introduction

The high tech sector and other more traditional industrial sectors are comprised of small and medium enterprises(SMEs) as well as large scale enterprises. For some context, within the European Union, SMEs are divided into three categories: Medium enterprises with fewer than 250 employees and less than 50 million in turnover, small enterprises with fewer than 50 employees and less than 10 million in turnover, and lastly Micro enterprises, with fewer than 10 employees and less than 2 million in turnover. Large enterprises are naturally the ones that have more employees and turnover than a Medium enterprise[sme16]. According to the 2016-2017 Annual SME report by Eurostat, there were 23,849 active non-financial SMEs, and 45 large enterprises, that together combined for almost 150,000 new job position in the specific year. So we understand that SMEs and Large enterprises play a significant role in the European Economy[PM17]. When we combine this fact with the rapid evolution of the technology and hence the technological background of their products and services, the systematic effort towards Cloud Computing adoption is underlined.

Moreover, in the very competitive business environment, enterprises need to maximize the efficiency of their product/service in order to stay relevant. As Cloud Computing services consumers, enterprises are attracted by the opportunity for reducing or eliminating costs associated with ”in-house” provision of their IT infrastructure. At the same time, since Cloud applications may be crucial to their core business operations, it is essential that the Cloud Service Providers(CSPs) provide guarantees on service delivery. Typically, these are provided through Service Level Agreements (SLAs) brokered between the CSPs and the consumers[Buy09]. This, in turn, creates a slew of factors that directly or indirectly affect the enterprises' decision of initially adopting Cloud Computing services, and secondary the specific Cloud configuration/architecture to fit their business model.

In this paper we are comparing how the factors, that are affecting the Cloud Computing adoption by varying size enterprises, have changed over the years. This comparison can reveal whether or not over the years, the perceived benefits and drawbacks of Cloud Computing adoption have remained the same, and if yes where the focus of the CSPs could be focused in the development/improvement of Cloud technologies. Also, this type of extended literature review can be preliminary work towards a ”universal model” of Cloud Computing adoption or migration of enterprise IT infrastructure. The points expressed above lead us to form the following research questions (RQ):

- *RQ1: How do the Cloud Computing adoption factors change over the years?*
- *RQ2: Do SMEs and large enterprises focus on different factors for Cloud Computing adoption?*

In Section 2 we are presenting a brief literature review, by mentioning the most important findings of the papers included in our paper set. Section 3 contains the data we extracted from analyzing our sources, as well as charts that further organize and compare it. In Section 4 we present the results of our analysis. Section 5 contains our conclusion, and the last Section 6, the limitations of this paper and future research suggestions.

## 2 Literature Review

According to a Forrester report that was analyzed by Forbes [Col17], the total global public Cloud market will be $178B in 2018, up from $146B in 2017, and will continue to grow at a 22% compound annual growth rate (CAGR). It also predicts that more than 50% of global enterprises will rely on at least one public Cloud platform to drive digital transformation and subsequently satisfy the customers' needs. This last figure, can only emphasize the need for extensive research in the field of Cloud Computing adoption. Although there has been satisfactory amount of relevant research, Asatiani [Asa15] argues that it shows a notable under-representation of environmental and organizational adoption determinants in the results.

Stemming from this idea, we chose our main sources to span the majority of the lifetime of Cloud Computing. Furthermore, sought to use a combination of survey based studies which are employing survey/interview methods to extract the perception of IT professionals, of varying experience, concerning Cloud Computing adoption at the time of the study. To supplement the survey based, we considered studies that performed extensive review of the available literature available at that time. All the surveying studies are also employing one or more theoretical frameworks in order to either form their conceptual model, or to validate and organize their results. Statistical analysis was also used for the aforementioned purpose, by some of the sources.

### 2.1 Introducing Analysis Methods

In this subsection, we briefly describe the main methods employed by the sources we examined.

- The Diffusion of Innovation(DOI) theory was developed in 1962, and it is used to explain how, over time, an idea or product gains momentum and diffuses (or spreads) through a specific population or social system. It focuses on 5 adopter categories (Innovators, Early Adopters, Early Majority, Late Majority, Laggards) and examines 5 factor that influence the adoption of innovation(Relative Advantage, Compatibility, Complexity, Triability, and Observability)[Rog62].

- Technology-Organization- Environmental(TOE) is a framework that identifies three context groups: The technological context refers to internal and external technologies applicable to the firm, the organizational context which refers to several indexes regarding the origination, such as firm size and scope, the complexity of managerial structure etc., and the environmental which refers to an enterprise's industry, competitors and government policy or intention. The TOE framework also explains the adoption of innovation and a considerable number of empirical studies have focused on various IS domains[TFC90].
- Technology Acceptance Model(TAM) was proposed by Fred Davis [Dav89], according to whom, Perceived Usefulness(PU) which defined as the degree to which an individual in the organization believes that using the technology will improve their performance and Perceived Ease Of Use (PEOU), which is defined as the extent to which a person believes that using a system will be free from effort, are two important factors which influence the decision to accept new technology [Dav89].

A summary of the occurrences of the employed frameworks and methods is presented in Table 1.

Table 1. Methods used throughout main sources

| **Methods** | DOI | TOE | TAM | Extensive Literature Reviewing | Surveying |
|---|---|---|---|---|---|
| **Occurrences** | 2 | 2 | 1 | 3 | 4 |

Briefly we can summarize the main sources that explicitly refer to factors of Cloud Computing adoption of our paper set, as follows:

1. The first source we analyzed, was published in 2011, and employed surveying enterprises that belonged in top 500 firms in the high-tech industry in Taiwan. Using a predetermined questionnaire and the Technology-Organization-Environment(TOE) framework they were able to conclude that the relative advantage of Cloud Computing services implementation could improve speed of business communications, efficiency of coordination among firms, customer communications, and access to market information mobilization. However, relative advantage was observed to have a significantly negative influence on Cloud Computing adoption of enterprises, in the high-tech industry. On the other hand, complexity and compatibility were not found to be the significant discriminators. Furthermore top management support and firm size were significant discriminators between Cloud Computing adopters and non-adopters. Whereas, technological readiness did not significantly impact Cloud Computing adoption. Lastly it was found that competitive and trading partner pressure were statistically significant for Cloud Computing adoption in the high-tech industry. Trading partner power also has a positive effect on IT adoption decisions[LCW11].
2. The next study was also conducted in Taiwan-based organizations. The purpose of this study, is to examine how Cloud Computing is understood by IT professionals who had at least three years of experience. The study used a survey of interviews in order to understand IT professionals' concerns about Cloud Computing. The findings of this study suggest that while the benefits of Cloud Computing such as its computational power and ability to help companies save costs are often mentioned in the literature, the primary concerns are that IT managers and software engineers have compatibility of the Cloud with companies' policy, IS development environment, and business needs; and relative advantages of adopting Cloud solutions. The findings also suggest that most IT companies in Taiwan will not adopt Cloud Computing until the concerns about Cloud Computing ,such as security and standardization are reduced and successful business have emerged[LC12].
3. Continuing with survey based studies, the following one was conducted by surveying 211 SMEs in the APAC (Asia-Pacific) region. The data was then analyzed among other techniques, using SmartPLS to build, run and validate the process model. They formulated 15 hypotheses by combining the five following core factors that stemmed from their literature review process: Cost reduction, convenience, reliability, sharing and collaboration and finally security and privacy. In an unexpected result, ease of use and convenience is a top priority for SMEs when it comes to Cloud Computing adoption, followed by security and privacy with cost reduction being the last of the positive factors. Reliability of Cloud Services affects the decision for Cloud Computing adoption, since the survey data suggested that it is not as reliable as it should be for their business. Sharing and collaboration also acts negatively on the decision, since SMEs tend to prefer more traditional means of communicating e.g. face to face meetings, phone calls etc. Some secondary positive factors were also mentioned such as cost efficiency, no need for extra office space nor the need to carry storage devices. Service is scalable in fast manner and faster content delivery is achievable[GSR13].
4. The extensive literature review[EG14] sources 51 articles and studies, and although it does not clearly provide a list of positive and negative factors for Cloud adoption, it succeeds in providing an extensive categorization of them. The initial separation, divides the factors into Cloud Computing Adoption factors and processes. The factors are further divided into External(Government regulations, IT industry standards institute, Cloud providers, Business partners etc) and Internal( Willingness to invest, top management, firm size, organizational culture etc). The Cloud Computing adoption factors are divided into the following categories: Evaluation, proof of concept, adoption decision, Implementation and integration, IT governance, and confirmation. Each of the aforementioned categories pertains to several more specific determinants, that affect the Cloud Computing adoption process[EG14]. A more detailed overview of these determinants is presented in the following chapter.
5. Asatiani's literature review manages to source 76 articles. By doing that he is able to first divide Cloud Computing adoption determinants into five categories: Drivers of adoption, inhibitors of adoption, CSPs, Organization, and external environment. Then he is presenting the percentage of the cases that each factor is either positively significant or negatively significant, and uses a simple system to show this difference. Moreover he identifies under-represented factors and issues in the Cloud Computing adoption literature[Asa15].
6. Our penultimate main source, is a study conducted in 2016, surveying and it offers the perspective of organizational flexibility. They surveyed 21 IT professionals with fourteen years of managerial experience on average, and whose enterprises have adopted Cloud Computing for at least 2 years. The study also utilized the Techonology-Organization-Environment framework, the Diffusion Of Innovations method and Technology Acceptance Method in order to retrieve and construct a conceptual model. This model included the following factors: Relative advantage, perceived usefulness, perceived ease of use, vendor credibility, support from top management, and organizational flexibility. Based on the survey answers, they were able to further divide organizational flexibility, into economic, process, performance, and market flexibility[LB16].

7. The most recent main source we analyzed was conducted in Germany in 2017. They interviewed ten IT professionals with at least 10 years of experience in IT management in specifically Large enterprises. The study also managed to survey IT professionals that at the time of the study were working in 7 different enterprises, ranging from large multinational IT consulting firms, to SMEs active only in Germany. They were able to assign a high or low importance, as well as positive or negative influence, when it comes to Cloud Computing adoption decision. Finally they were able to list the biggest differences of these factors between Large enterprises and SMEs. They conclude that the factors that are common between SMEs and large enterprises, but have different influence on the decision for adopting Cloud Computing are the following: Lack of interoperability, switching cost, lack of customization, data protection issues, security issues, existing installations, taxation issues.[KML17]

## 3 Factor Identification and Data Acquisition

Upon analyzing the main sources of our paper set, we were able to document the various factors for Cloud Computing adoption in the context of yearly intervals and the size of the enterprise. We set out to consolidate and organize the factors that may lead an enterprise to adopt Cloud Computing solutions in order to examine which factors remain constant over time and which factors are added from year to year. Furthermore we can easily examine which factors are similar between Large Enterprises and SMEs, avoiding at the same time risk of overlapping. Table 3 contains the retrieved data and it is organized vertically in regard to the year the study was conducted, and horizontally on whether or not they were found to be applicable on SMEs or Large enterprises. For completion, we also included a plus sign or a minus sign next to each of them, in order to express whether or not they were found to have an explicitly positive or a negative impact on the enterprises' decision.

Afterwards we characterize the nature of each Cloud Computing adoption factor, using one or more of the following categories:

- Business(B): This category describes Cloud Computing adoption factors that pertain to the Business activities of the enterprise, e.g., the product viability, business relations, opportunities for investment etc.

- Technical(T): This category describes Cloud Computing adoption factors that pertain to the technical side of Cloud Computing adoption, e.g., required IT skills, need for maintenance of the infrastructure, existing installations etc.

- Organizational(O): This category describes Cloud Computing adoption factors that pertain to the internal structure of the enterprise, e.g., human resources, firm size etc.

- Financial(F): This category describes Cloud Computing adoption factors that pertain to the finance of an enterprise, e.g., influencing profits, required investments etc.

- Regulatory(R): This category describes Cloud Computing adoption factors that pertain to the regulations surrounding the use of Cloud Computing on an enterprise level, and can include government regulations, CSPs regulation, legal regulations regarding data protection of the enterprise's customers etc.

We choose the aforementioned categories, in order to provide a frame of reference that is spherical enough to be able to describe many aspects of an enterprise's main focus.
Each factor can belong in many of the aforementioned categories. This assessment is a combination of the definition of the Cloud Computing adoption factors contained in our sources, and our intuition and experience. It is worth noting that one could argue that on an enterprise level, the Cloud Computing adoption factors we identified are influencing all five of the aforementioned aspects of an enterprise. We choose to document only the direct implications that a factor might contribute to or alleviate.

We constructed Table 4 in which, the aforementioned categorization can be seen. Also we grouped relevant factors together, in order to minimize the overlap that is a result of our sources using different names for similar concepts. The factors are presented in the first column, while in the parentheses one can find the factors that were grouped with it. The second column contains a unique identifier to refer to each of the factors, and it is used in the accompanying graphs, in order to make them reader friendly. The third column presents the categories that each factor pertains to. The last four columns, contain the amount of times the respective factor was found to have a positive, negative or neutral impact to the enterprises Cloud Computing adoption. The cases of positive and negative impact were explicitly mentioned in our sources. The case of neutral impact can mean two things. Either the source found the factor in question to literally have neutral impact on Cloud Computing adoption, or the source merely identifies the Cloud Computing adoption factors that influence the enterprise's decision but it does not assign a positive or negative impact to it. From Table 4 we constructed the following chart.

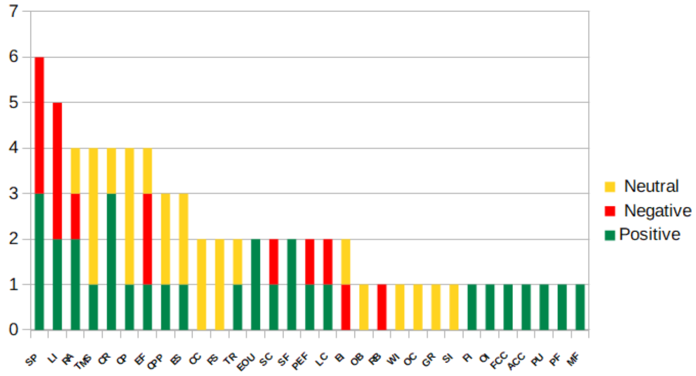


Fig. 1. Accumulated impact of Cloud Computing adoption factors

The reader can directly compare the discovered impact of the factors on Cloud Computing adoption factors over the years. The factors are presented in the order we document them in Table 4 but can also use the identifiers along the X-axis.

We sum up the aforementioned categorization into Table 3, but in terms of the size of the enterprise. We normalize our findings, in order to compare them to each other, regardless of the amount of the factors we identified for each enterprise size.

| Categories | SMEs | Large Enterprises |
|---|---|---|
| Business | 9 (21.4%) | 12 (21.1%) |
| Technical | 11 (26.2%) | 15 (26.3%) |
| Organizational | 10 (23.8%) | 14 (25%) |
| Financial | 10 (23.8%) | 13 (22.8%) |
| Regulatory | 2 (4.8%) | 3 (5.3%) |

Table 2. Categorization of the Cloud Computing adoption factors of enterprises of different size

In order to provide a visual representation of the contents of Table 3, we constructed Figure 2. The reading of this figure is quite intuitive, and the findings are discussed in the following section.

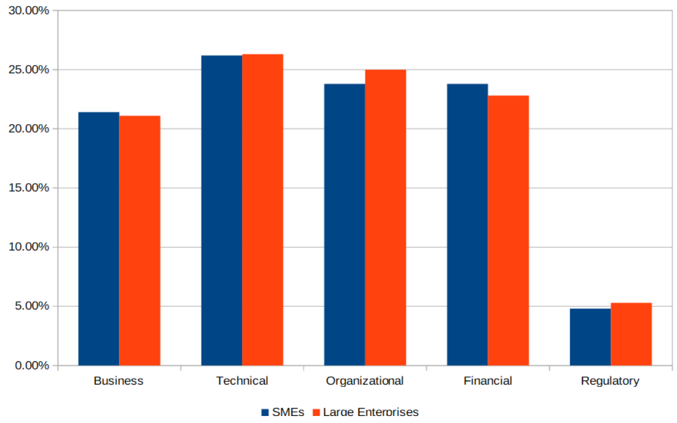


Fig. 2. Percentage of categories that Cloud Computing adoption factors pertain, in enterprises of different size.

## 4 Results

In this section we discuss the results obtained from the previous section, and try to relate them to the research questions stated in the introduction of the paper. Initially we set out to answer whether or not the Cloud Computing adoption factors have been changing throughout the years(RQ1). Our results show that depending on the point of view of the research paper in question, we get similar adoption factors. Namely, papers that employ surveying, tend to present same or equivalent factors that affect Cloud Computing adoption. From Table 4 we can see that though out the period in question, several factors were mentioned more frequently than others. Specifically, regardless of the positivity or negativity of their impact, factors such as Relative Advantage, Top Management Support, Security and Privacy, Cloud providers, Cost Reduction, etc, keep coming up in the results of research papers. We observe a shift after 2015, as the papers from 2015 and later, offer findings that are quite extensive and more importantly very specialized, to either the size of the enterprise, or in the case of Asatiani[Asa15] paper which is a very thorough literature review. Specifically in 2016 and 2017, we observe the use of different points of view, in order to inspect the impact of Cloud Computing adoption in the organizational flexibility of a large enterprise, or distinguish the impact of certain factors, on enterprises of different sizes respectively.

Futhermore, we aggregate the categories that impactful factors of Cloud Computing adoption pertain to. This comparison yielded surprising results, as Figure 2 shows, the categories of factors, have a very small variance between SMEs and Large enterprises. Namely the difference in Business related factors, could stem from the fact that SMEs are more specialized compared to Large enterprises, have smaller budgets, their income/profit sources are less diverse and structural mistakes or a couple bad investments, can put them out of business.

The percentage of Technical related factors, is almost the same for SMEs and Large enterprises. They are slightly more important for the latter, since adopting a new technology as complex as Cloud Computing, across the entire structure of the enterprise can be challenging. If we refer back to the definition of Large enterprises, we can see that is especially difficult to implement such a complex change as cloud adoption, since they are at least 250 employees( usually employee number is in the thousands). Implementing an infrastructure that is meant to be used by this amount of persons at the same time, is a technical challenge.

The above comparison can be referenced when we observe the biggest difference in our comparison, that of organizational related factors. The changes in large enterprises are more difficult to implement for the aforementioned reasons, and our result suggests this.

| Year | Factors | |
|---|---|---|
| | SMEs | Large Enterprises |
| 2011 | - | Relative Advantage(-)<br>Complexity & Compatibility<br>Top Management Support<br>Firm Size<br>Technology Readiness<br>Competitive & Trading Partner Pressures |
| 2012 | - | Relative Advantage<br>Compatibility<br>Complexity<br>Observability<br>Trialability |
| 2013 | Cost Reduction(+)<br>Ease of Use & Convenience(+)<br>Reliability(-)<br>Sharing & Collaboration(-)<br>Security & Privacy(+) | - |
| 2014 | - | Willingness to Invest<br>Top Management<br>Firm Size<br>Organization Culture<br>Employees IT skills<br>Government Regulations<br>IT Standards Institutes<br>Cloud Providers<br>Business Partners<br>Cloud Service Broker |
| 2015 | Cost Advantage(+)<br>Fast Implementation(+)<br>Opportunities for Innovation(+)<br>Strategic Flexibility(+)<br>Focus on Core Competences(+)<br>Accessibility(+)<br>Trialability(+)<br>Relative Advantage(+)<br>Online Collaboration(+)<br>Management Support(+)<br>Attitudes towards Technology(+)<br>Security & Privacy(-)<br>Performance Risks(-)<br>Economic Risks(-)<br>Lock-In(-)<br>Providers Reputation(+)<br>Partner Pressure(+) | - |
| 2016 | - | Relative Advantage(+)<br>Perceived Usefulness(+)<br>Perceived Ease of Use(+)<br>Vendor Credibility<br>Top Management support<br>Organizational Flexibility(+)<br>Economic Flexibility(+)<br>Process Flexibility(+)<br>Performance Flexibility(+)<br>Market Flexibility(+) |
| 2017 | Lack of Interoperability(+)<br>Switching Cost(+)<br>Lack of Customization(+)<br>Data Protection Issues(+)<br>Security Issues(+)<br>Existing Installations<br>Taxation Issues<br>Cost Reduction(+) | Lack of Interoperability(-)<br>Switching Cost(-)<br>Lack of Customization(-)<br>Data Protection Issues(-)<br>Security Issues(-)<br>Existing Installations(-)<br>Taxation Issues(-)<br>Cost Reduction |

Table 3. Identified Cloud Computing adoption factors

The second to last comparison refers to the Financial related factors. By drawing similar comparisons as we do in the Business related factors, we can see that SMEs are more cautious when they make investments. Also cost reduction is a factor that is very desirable by SMEs throughout our research.
Lastly, the Regulatory factors, naturally affect Large enterprises more than SMEs. Many of the Large enterprises are multinational companies, having business interests all around the globe. This introduces national and international regulations that enterprises should be aware of and abide to. This includes taxation regulations, employment regulations, as well as data protection regulations.
Even though the differences are small, RQ2 is answered in an unsurprising manner. We believe that with a bigger data sample, these differences will become more concrete, and even grow bigger.

## 5 Conclusion

In this paper we surveyed the factors that play a significant or insignificant role for different size enterprises in order to adopt Cloud Computing solutions. We presented a brief literature review of the papers chronologically in order to provide a better understanding about the topic. Also we discussed the methods that the papers used for evaluating or rejecting their hypotheses and we summarized in Table 3 all the factors that affect Cloud Computing adoption. Finally we provide the results of our study in order to answer the research questions we provided in the Introduction.
Our conclusion is twofold. According to our research for RQ1, we came up to the inference that the main point of change over the years are not the factors that affecting Cloud Computing adoption but the perception for these factors. Furthermore we observed an evolution when it come to the Cloud Computing adoption factors, that can be attributed to two points. Firstly, during the early stages of Cloud Computing adoption research, we see a more aggressive overlap when it comes to the retrieved factors. This is a result of the immaturity of Cloud Computing and the relevant research of the time. The majority of the enterprises have not realized the benefits of Cloud Computing and their customers didn't demand products that are cloud related[LC12]. As the technology matures we see more elaborate studies, focusing on the organizational and environmental factors that might affect the decision of adopting Cloud Technologies.
Furthermore, from Figure 3 we concluded that there are only small differences between the factors that enterprises take into consideration in order to adopt Cloud Solutions or not, regardless of their size.

## 6 Limitations and future research

One serious limitation of this study, is the relatively short amount of time we had in our disposal to carry out significant literature review and research. The paper set we constructed focused on use cases mainly in Asia, which might convince us that Cloud Computing is more utilized in this specific area, a claim that has not been proven in this paper. Concerning the paper set, it might have been wiser to choose case studies equally spread around the globe.
When it comes to the future research, according to Asatiani[Asa15], Cloud Computing is now entering its mature stage, so we can expect an influx of research. Especially interesting would be to research the government and international regulations when it comes to the usage of Cloud Computing. Also following the example of Lin and Chen[LC12]. Future research could focus on how these factors are correlated with each other. In this case a model for Cloud Computing adoption or Migration could be starting to form. Also, it would be interesting to repeat the process of our paper in 3 or 5 years, using a bigger paper set, in order to verify our current conclusions.

Table 4.

| Factors | Identifier | Category |
|---|---|---|
| Relative Advantage | RA | B, T, F |
| Complexity & Compatibility | CC | T, O |
| Top Management Support | TMS | B, O |
| Firm Size | FS | B, O |
| Competitive & Trading Partner Pressures (Business Partners) | CPP | B, F |
| Observability | OB | B, T, O, F |
| Trialability | TR | T |
| Cost Reduction (Cost Advantage) | CR | F, B |
| Ease of Use & Convenience ( Perceived Ease Of Use ) | EOU | B, O |
| Reliability | RB | T |
| Sharing and Collaboration (Online Collaboration) | SC | O, B |
| Security and privacy ( Data Protection Issues) | SP | T, O, F |
| Willingness to Invest | WI | O, F |
| Organization Culture | OC | B, O |
| Employees IT skills (Attitude towards Technology, Technology Readiness) | ES | T, O, F |
| Government Regulations | GR | R, B |
| IT Standards Institutes | SI | R |
| Cloud Providers ( Cloud Service Broker, Providers Reputation, Vendor Credibility) | CP | T, F |
| Fast implementation | FI | T |
| Opportunities for Innovation | OI | B, F |
| Strategic Flexibility (Organizational Flexibility) | SF | B, O |
| Focus on Core Competences | FCC | B, O, F |
| Accessibility | ACC | O, T |
| Lock-In ( Switching Cost, Lack of Interoperability) | LI | T, F |
| Perceived Usefulness | PU | O, T |
| Economic Flexibility (Taxation Issues, Economic Risks ) | EF | F, R |
| Process Flexibility | PF | T |
| Performance Flexibility (Performance Risks) | PEF | T, B, O |
| Market Flexibility | MF | B, F |
| Lack of customization | LC | T |
| Existing Installations | EI | T, O |

## References

[Asa15] Aleksandre Asatiani. Why cloud? -a review of cloud adoption determinants in organizations. 05 2015.

[Buy09] Rajkumar Buyya. Market-oriented cloud computing: Vision, hype, and reality of delivering computing as the 5th utility. *2009 9th IEEE/ACM International Symposium on Cluster Computing and the Grid*, 2009.

[Col17] Louis Columbus. Forrester's 10 cloud computing predictions for 2018. https://www.forbes.com/sites/louiscolumbus/2017/11/07/forresters-10-cloud-computing-predictions-for-2018/250558984ae1, Nov 2017.

[Dav89] Fred D. Davis. Perceived usefulness, perceived ease of use, and user acceptance of information technology. *MIS Quarterly*, 13(3):319, 1989.

[EG14] Rania El-Gazzar. *Proceedings of the 4th International Conference on Cloud Computing and Services Science*, 2014.

[GSR13] Prashant Gupta, A. Seetharaman, and John Rudolph Raj. The usage and adoption of cloud computing by small and medium businesses. *International Journal of Information Management*, 33(5):861874, 2013.

[KML17] Surya Karunagaran, Saji K Mathew, and Franz Lehner. Differential cloud adoption: A comparative case study of large enterprises and smes in germany. *Information Systems Frontiers*, 2017.

[LB16] Prerna Lal and Sangeeta Shah Bharadwaj. Understanding the impact of cloud-based services adoption on organizational flexibility. *Journal of Enterprise Information Management*, 29(4):566588, Nov 2016.

[LC12] Angela Lin and Nan-Chou Chen. Cloud computing as an innovation: Percepetion, attitude, and adoption. *International Journal of Information Management*, 32(6):533540, 2012.

[LCW11] Chinyao Low, Yahsueh Chen, and Mingchang Wu. Understanding the determinants of cloud computing adoption. *Industrial Management Data Systems*, 111(7):10061023, 2011.

[PM17] Jenna Julius Daniel Herr Dimitri Gagliardi Chiara Marzocchi Olivia-Kelly Lonkeu Jill Wenger Patrice Muller, Paula Ramada. Annual report on european smes 2016/2017: Focus on self-employment - working paper. Technical report, 2017.

[Rog62] Everett Rogers. Diffusion of innovation theory. goo.gl/pJHLRw, 1962.

[sme16] Eurostet statistics explained, enterprise size. goo.gl/2TmEMb, 2016.

[TFC90] Louis G. Tornatzky, Mitchell Fleischer, and Alok K. Chakrabarti. *The processes of technological innovation*. Lexington, 1990.

university of groningen

faculty of science
and engineering

computing science