VOL 98 No 2
June 2007

SAIEE Africa Research Journal

SAIEE Africa Research Journal
Research Journal of the South African Institute of Electrical Engineers
Incorporating the SAIEE Transactions
June 2007
Volume 98 No. 2

GUEST EDITORIAL

PATTERN RECOGNITION: PART I

Three special issues of the SAIEE Africa Research Journal, entitled Pattern Recognition: Part I, Pattern Recognition Part II and Pattern Recognition: Part III, are devoted to selected papers from the Pattern Recognition Association of South Africa's 2006 workshop held in Parys, South Africa from 29 November to 1 December 2006. This workshop was peer-reviewed and each paper was reviewed by at least two reviewers. Reviewers could recommend a reviewed paper to the technical chairs for publication in this special issue of the SAIEE Africa Research Journal. The selection process consisted of two rounds, the first round of which formed the normal review process for papers submitted to be considered for PRASA 2006. The second round applied only to papers that were recommended for journal publication. The second round of screening served the purpose of requesting authors to update their manuscripts and for reviewers to verify that all requests have been addressed. A total of 67 papers (44 presentations and 23 posters) were accepted for the workshop. A total of sixteen papers passed the second round of the review process for publication in these special issues.

The first six papers appear in this special issue, Volume 98 Number 2 of the SAIEE Africa Research Journal, Pattern Recognition: Part I. The remaining ten papers will appear in Volume 98 Numbers 3 and 4 of the SAIEE Africa Research Journal, Pattern Recognition: Part II. Parts I and II contain a diversity of papers in the field, whereas Part III focuses exclusively on Human Language Technologies.

Subdivision schemes are widely used in various applications such as data fitting, computer graphics and solid modelling, to name a few. In the first paper in this issue, entitled *Subdivision of Curves and Surfaces: An Overview*, Herbst, Hunter and Rossouw demonstrate the basic ideas of subdivision schemes for curves. They consider both interpolatory and corner-cutting schemes as well as their adaptation to finite sequences. Some specific examples of surface subdivision are also discussed.

The problem of automated classification of froth into different classes as part of process control in chemical plants, in many different applications, has been receiving ever more attention during the past few years. In their paper entitled *Unsupervised Classification of Dynamic Froths* Forbes and de Jager show that unsupervised classification algorithms can be used to automatically detect a user specified number of froth classes. Some interesting relationships between froth structure and statistical indicators are highlighted based on their experimental results.

Dealing with missing data in data sets is a problem that researchers frequently run into. In *Fuzzy ARTMAP and Neural Network Approach to Online Processing of Inputs with Missing Values*, Nelwamondo and Marwala propose an ensemble-based approach for dealing with missing data, without prediction or imputing the missing values. This scheme is suitable for online operations of neural networks and hence well suited for online applications, for example, online condition monitoring. The results obtained from the comparative study show that the proposed technique performs better on regression problems.

Active appearance models provide an elegant framework for tracking objects. However, using them in a deterministic algorithm to perform tracking is not robust enough since no history is used of the object's movement and position. In their paper entitled *On Visual Object Tracking Using Active Appearance Models* Hoffman, Herbst and Hunter present two approaches to rectify this problem. Both techniques are based on particle filters. Their experimental results indicate the effectiveness of the proposed schemes.

In their paper, *Identity Confidence Estimation of Maneuvering Aircraft*, Holtzhausen and Herbst apply multiple hypothesis techniques to extract an identity confidence from a track, given a set of possible tracks and observations. Their system utilizes numerous estimation filters internally which are investigated and compared. The authors present results obtained from a radar simulation system as well as from a series of benchmark tests.

Part I of the Special Issue on Pattern Recognition concludes with a brief communication entitled *A Note on Difference Spectra for Fast Extraction of Global Image Information* in which the authors van Wyk, van Wyk and van den Bergh demonstrate the use of the concept of an Image Difference Spectrum as a fast alternative to pattern spectra derived from computationally intensive granulometric techniques. This means that granulometric analyses can now be performed in real-time due to the low computational complexity of the method.

MA van Wyk and BJ van Wyk
Guest Editors

SUBDIVISION OF CURVES AND SURFACES: AN OVERVIEW

Ben Herbst, Karin M Hunter, Emile Rossouw

Applied Mathematics, University of Stellenbosch, Private Bag X1, Matieland, 7602, South Africa

Abstract: Subdivision schemes are widely used in various applications such as data-fitting, computer graphics and solid modeling. In this paper we present the basic ideas of subdivision schemes for curves; both interpolatory and corner-cutting schemes, as well as their adaptation to finite sequences. We conclude with examples of specific applications for these subdivision schemes and provide an example of surface subdivision.

Key Words: subdivision, curve fitting, approximation theory

1. INTRODUCTION

Few people fail to be impressed by the quality of the graphics of recent animated movies. Geri's Game by Pixar is a beautiful example. The keen observer will note the names of Edwin Catmull and Jim Clark among the credentials. Their contributions to animation, based on their subdivision scheme for surfaces, won them an Academy Award for Technical Achievement in 2006.

To understand what subdivision is all about, one should realise that the quality of the three-dimensional graphics depend, among others, on the modeling of the objects themselves. As in the case of Geri, one wants to construct a face based on a limited number of control points that defines the basic structure. This implies that given the control points, the region between the control points should be constructed in a realistic way. One can of course use interpolation, typically spline interpolation in which case the interpolant is first constructed and then evaluated at the required points. Should one decide to move one of the control points the process starts all over. Subdivision schemes skip the first step of constructing the interpolant. Instead it proceeds directly from the control points to the filled-out surface through an iterative procedure. Moreover, the process is local with the advantage that any change in control points have only a local effect. Changing Geri's nose by moving a control point does not for example, affect his mouth. Apart from computer animation, subdivision schemes also find wide applications in computer graphics and solid modeling.

In this paper we explain the basic ideas behind subdivision schemes on curves before we briefly indicate how these ideas carry over to the subdivision of surfaces.

Suppose we are given a bi-infinite sequence of points $c^{(0)} = \{c_j^{(0)} : j \in \mathbb{Z}\}$, and are interested in approximating these points with a smooth curve. Standard ways of doing this involves constructing an approximating function, e.g. an interpolating spline. Then, in order to visually represent the approximating function in computer applications, the function needs to be evaluated on a sufficiently dense set of points. Subdivision schemes skip the first step by creating a dense set of points directly from the given points, i.e. there is no need to first construct the approximating function and then evaluate it. This leads to considerable savings in computational cost.

Of course, if the original points contain noise, the approximating curve should not pass through them, i.e. the approximating curve should not be interpolatory. This can be achieved in different ways: It is possible to first apply a smoothing operation to the given points and then do an interpolation, or one can use something like a smoothing spline. In this paper we describe two techniques of subdivision, one interpolatory and one smoothing (or corner-cutting).

Consider the following simple iterative procedure: Start with a set of points $c^{(0)}$, called the *original control points*, and generate a new set of control points $c^{(1)} = \{c_j^{(1)} : j \in \mathbb{Z}\}$ by taking a linear combination of the original control points. Repeat this until the desired density is achieved.

For example if one generates the new control points using the simple linear combination

$$c_{2j}^{(1)} = c_j^{(0)} \text{ and } c_{2j+1}^{(1)} = \tfrac{1}{2}(c_j^{(0)} + c_{j+1}^{(0)}), \quad j \in \mathbb{Z}, \quad (1)$$

then the even-indexed elements of the new control points are simply the original points and the odd-indexed elements are generated halfway between the old control points. This step is then repeated indefinitely, roughly doubling the number of points at each step. In this case the points fill in or converge to the straight-line segments connecting the original control points, as illustrated in Figure 1. Thus we obtain a continuous piecewise linear curve. This simple procedure is an example of a subdivision scheme.

In general, given a sequence $a = \{a_j : j \in \mathbb{Z}\}$, called

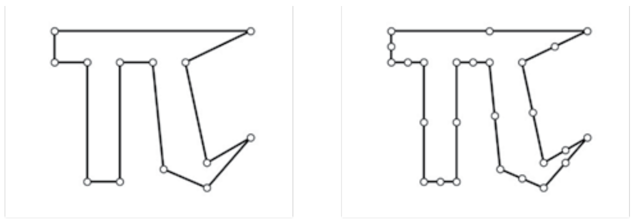

(a) original control points *(b) control points after one iteration*

Figure 1: Illustration of the iterative procedure Equation 1.

the *mask* of the subdivision scheme, and a sequence of control points $c = \{c_j : j \in \mathbb{Z}\}$, we define the corresponding subdivision operator S_a by

$$(S_a c)_j = \sum_{k\in\mathbb{Z}} a_{j-2k} c_k, \quad j \in \mathbb{Z}. \tag{2}$$

The resulting subdivision scheme is then defined, for a given sequence of control points c, by

$$c^{(0)} = c, \quad c^{(r+1)} = S_a c^{(r)}, \quad r = 0, 1, \ldots, \tag{3}$$

or, equivalently,

$$c^{(0)} = c, \quad c^{(r+1)} = S_a^{r+1} c, \quad r = 0, 1, \ldots. \tag{4}$$

Note that the nature of the subdivision scheme is entirely determined by the choice of the mask, i.e. the linear combination used to generate the new control points at each iteration. The key therefore is to find an appropriate mask. The choice of the mask determines (i) whether the subdivision scheme is interpolatory or smoothing (corner-cutting), (ii) the convergence of the scheme, and (iii) the smoothness of the final curve.

These issues are closely related to the existence of a *refinable function*, as briefly discussed in the sections below, and refinable functions provide a link to wavelets, see e.g. [1].

There is no unique or best way of choosing a mask. One possibility is to demand that if the original control points fall on a polynomial of a certain degree, then the newly generated control points must lie on the same polynomial. This is, in fact, the idea behind Dubuc-Deslauriers subdivision scheme [2, 3]. In Section 2, we develop this idea to derive explicit formulae for the Dubuc-Deslauriers mask and provide an adaptation for finite control sequences.

In Section 3 we introduce corner-cutting subdivision schemes and their corresponding refinable functions, for infinite as well as finite sequences of control points.

A few applications of the subdivision of curves are given in Section 4 and surface subdivision is briefly discussed in Section 5.

2. INTERPOLATORY SUBDIVISION

In this section we introduce the well known Dubuc-Deslauriers subdivision scheme as an optimally local, curve filling iterative procedure that reproduces polynomials of a given odd degree. We then indicate how the limit curve for the Dubuc-Deslauriers scheme depends on the existence of an associated refinable function and provide an adaption of this scheme for finite sequences.

2.1. *Construction of the mask*

Suppose the original control points fall on a polynomial p of degree $2n-1$, i.e. $c_j^{(0)} = p(j)$, $j \in \mathbb{Z}$. Consider the problem of finding the shortest possible mask a such that all the subsequent iteratives $c^{(r)}$ fall on the same polynomial. More specifically, we require that

$$\sum_{k\in\mathbb{Z}} a_{j-2k} p(k) = p\left(\frac{j}{2}\right), \quad j \in \mathbb{Z}. \tag{5}$$

The mask is derived from the standard Lagrange polynomials of degree $2n-1$, uniquely defined by

$$\ell_k(j) = \delta_{k,j}, \quad k, j = -n+1, \ldots, n, \tag{6}$$

and therefore satisfying the polynomial reproduction property

$$\sum_{k=-n+1}^{n} p(k)\ell_k(x) = p(x), \quad x \in \mathbb{R}. \tag{7}$$

An explicit formula for these Lagrange functions, for $k = -n+1, \ldots, n$, is given by

$$\ell_k(x) = \prod_{\substack{k=-n+1\\ k\neq j}}^{n} \frac{x-j}{k-j}, \quad x \in \mathbb{R}. \tag{8}$$

Comparing Equation 7 (with $x = \frac{1}{2}$) and Equation 5 (with $j = 1$), it follows readily that the shortest possible mask satisfying Equation 5 is given by

$$\begin{aligned} a_{2j} &= \delta_{j,0}, & j \in \mathbb{Z} && \text{(9a)}\\ a_{2j+1} &= \ell_{-j}\left(\tfrac{1}{2}\right), & j = -n \ldots, n-1, && \text{(9b)}\\ a_{2j+1} &= 0, & \text{otherwise.} && \text{(9c)} \end{aligned}$$

This mask is known as the Dubuc-Deslauriers mask [3].

Note that Equation 9a implies that Equation 3 satisfies the interpolatory property

$$c_{2j}^{(r+1)} = c_j^{(r)}, \quad j \in \mathbb{Z}, \; r = 0, 1, \ldots, \tag{10}$$

i.e. the Dubuc-Deslauriers subdivision scheme is *interpolatory*. Also, the mask coefficients are symmetric, i.e.

$$a_j = a_{-j}, \quad j \in \mathbb{Z}. \tag{11}$$

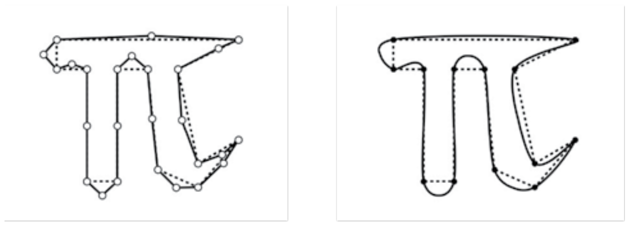
Therefore the Dubuc-Deslauriers subdivision scheme is interpolatory, symmetric and fills polynomials of degree $2n-1$.

For example, if $n=1$, Equation 9, Equation 2 and Equation 3 yield for $r=0$, the iteration procedure of Equation 1. This subdivision scheme converges to a continuous piecewise linear function that interpolates the original control points (see Figure 1).

For $n=2$ we get

$$a_{2j+1} = \begin{cases} -\frac{1}{16}, & j=-2, \\ \frac{9}{16}, & j=-1, \\ \frac{9}{16}, & j=0, \\ -\frac{1}{16}, & j=1, \\ 0, & \text{otherwise.} \end{cases} \qquad (12)$$

Subdivision with this mask converges to a smooth function [3], while still interpolating the original control points, see Figure 2.



(a) control points: original (●), after one iteration (○) *(b) original control points (●), control polygon after 6 iterations (–)*

Figure 2: Dubuc-Deslauriers subdivision for $n=2$

It is interesting to note that Knuth based his construction of TEX fonts [4, Chapter 2] on ideas remarkably similar to subdivision schemes more than 10 years before the Dubuc-Deslauriers scheme was introduced in [3].

Figure 2 suggests that the Dubuc-Deslauriers subdivision converges to a smooth function for $n=2$. For a proof see [3, 5]. This limiting curve is described in terms of a *refinable* function, to be discussed in the next section.

2.2. *Convergence of Dubuc-Deslauriers subdivision*

The mask a of a convergent subdivision scheme ensures the existence of a function ϕ satisfying

$$\phi(x) = \sum_{j\in\mathbb{Z}} a_j \phi(2x-j), \quad x\in\mathbb{R}. \qquad (13)$$

We call such a function an *refinable function.*

It is shown in [3, 5] that the Dubuc-Deslauriers mask a generates a convergent subdivision scheme, which then guarantees the existence of an associated refinable function ϕ. Moreover, the refinable function inherits the mask's finite support and symmetry, as well as its polynomial filling and interpolatory properties as follows

$$\phi(x) = 0, \qquad x \notin (-2n+1, 2n-1), \quad (14)$$
$$\phi(x) = \phi(-x), \qquad x\in\mathbb{R}, \quad (15)$$
$$\sum_{j\in\mathbb{Z}} p(j)\phi(x-j) = p(x), \quad x\in\mathbb{R}, \quad (16)$$
$$\phi(j) = \delta_{j,0}, \qquad j\in\mathbb{Z}, \quad (17)$$

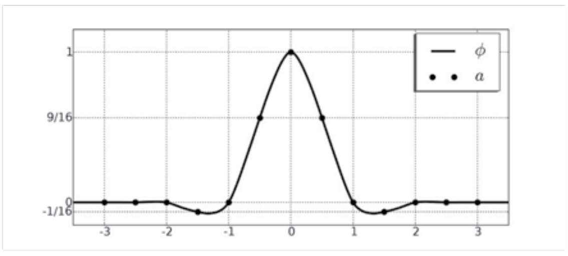
where p is any polynomial of degree $\leq 2n-1$. Also, the values of the refinable function at the half-integers are the values of the mask

$$\phi\left(\tfrac{j}{2}\right) = a_j, \quad j\in\mathbb{Z}. \qquad (18)$$

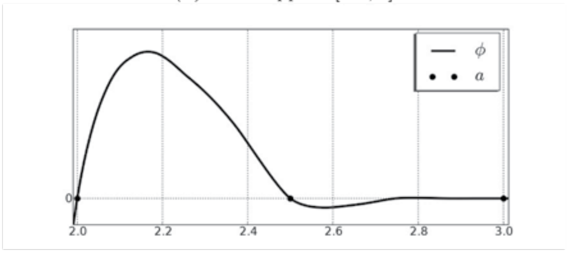
Finally, given the initial control points c, the limiting curve f of the Dubuc-Deslauriers subdivision scheme is given in terms of the refinable function as

$$f(x) = \sum_{j\in\mathbb{Z}} c_j \phi(x-j), \quad x\in\mathbb{R}. \qquad (19)$$

Some of these properties are illustrated in Figure 3 below.



(a) Full support $[-3,3]$



(b) Zoomed to $[2,3]$

Figure 3: Dubuc-Deslauriers refinable function ϕ and mask a.

The convergence of a subdivision scheme S_a ensures the existence of an associated refinable function since by choosing the original control points as the delta sequence $c=\delta=\{\delta_{0,j} : j\in\mathbb{Z}\}$ in Equation 4 the limit curve will be $f(x)=\sum_{j\in\mathbb{Z}}\delta_{0,j}\phi(x-j)=\phi(x),\ x\in\mathbb{R}$.

The converse of this statement is also true for interpolatory subdivision schemes. But for non-interpolatory subdivision schemes there are refinable functions for which the associated subdivision scheme is divergent, as shown in [6].

2.3. *A modified subdivision scheme for finite sequences*

The algorithms for bi-infinite sequences, as described in the previous sections, are applied mainly in the case of periodic sequences. For finite sequences these algorithms must be modified to accommodate the boundaries. Here we consider a method of adapting the Dubuc-Deslauriers subdivision scheme of Section 2.1 to the situation where the initial sequence c is finite.

The construction of the mask for finite sequences follows along similar lines as for the infinite case. The difficulty is that some of the values of the polynomial p in Equation 7 lie outside the finite domain and need to be supplied. This implies that an alternative mask needs to be constructed in the vicinity of the boundary. Following [7] and [8], we fit a polynomial of degree $2n-1$ to the $2n$ points next to, and including the boundary. Evaluating the resulting Lagrange polynomials at the half-integers next to the boundary yields the desired mask. The modified scheme for the left hand boundary $(j = 0, 1, \ldots)$ is given by

$$\begin{aligned} c_{2j}^{(r+1)} &= c_j^{(r)}, \\ c_{2j+1}^{(r+1)} &= \sum_{k\geq 0} a_{j,k} c_k^{(r)}, \end{aligned} \tag{20}$$

where for $j = 0, 1, \ldots, n-2$ (close to the boundary),

$$a_{j,k} = \ell_{k-n+1}\left(j + \tfrac{1}{2} - n + 1\right) \tag{21}$$

for $k = 0, \ldots, 2n-1$, and $a_{j,k} = 0$ for $k \notin \{0, 1, \ldots, 2n-1\}$. For $j \geq (n-1)$ (away from the boundary)

$$a_{j,k} = \begin{cases} \ell_{k-j}(\frac{1}{2}), & k = -n+1+j, \ldots, n+j, \\ 0, & \text{otherwise.} \end{cases}$$

If $n = 2$, for example Equation 21 gives

$$a_{0,k} = \ell_{k-1}(\tfrac{1}{2}) = \begin{cases} \frac{5}{16}, & k = 0, \\ \frac{15}{16}, & k = 1, \\ -\frac{5}{16}, & k = 2, \\ \frac{1}{16}, & k = 3, \\ 0, & \text{otherwise,} \end{cases}$$

and for $j \geq 1$ the mask is the same as before (see 12),

$$a_{j,k} = \begin{cases} -\frac{1}{16}, & k = 0, 3 \\ \frac{9}{16}, & k = 1, 2 \\ 0, & \text{otherwise.} \end{cases}$$

In the presence of a right hand boundary, the mask has to be modified in the same way as the left hand modifications, with the order reversed.

It is shown in [8] that a set of refinable functions exists for this modified mask (defined similarly to the definition Equation 13). The boundary modifications of the refinable function are illustrated in Figure 4. The existence of a set of refinable functions for the

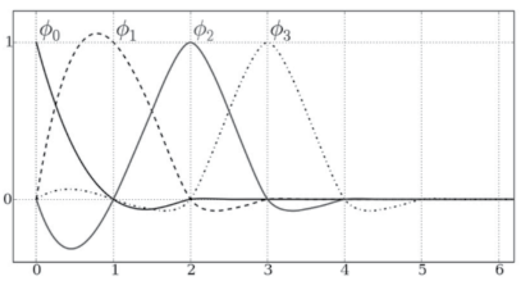


Figure 4: The refinable functions associated with the boundary-modified mask with $n = 2$

boundary-modified subdivision scheme allows one to construct wavelets for finite intervals. It is remarkable that these wavelets have finite decomposition and reconstruction sequences [8].

Next we discuss the so-called corner-cutting subdivision schemes.

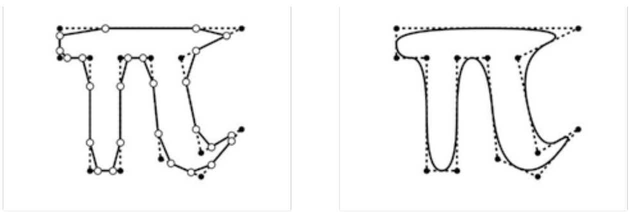
3. CORNER-CUTTING SUBDIVISION

In the case where the original control points contain noise we would not want to use an interpolatory subdivision scheme directly, but rather include some smoothing in the approximation. Since the limit curve of corner-cutting subdivision schemes does not pass through the control points, which amounts to some smoothing, corner-cutting subdivision schemes are better suited to this approximation problem. In this section we discuss one class of corner-cutting subdivision schemes, namely the de Rham-Chaikin scheme [9] and its generalization, the Lane-Riesenfeld scheme [10].

The only difference between the corner-cutting and the interpolatory schemes discussed in Section 2 lies in the choice of the mask of the operator in Equation 2. Conceptually, masks with positive entries generate corner-cutting subdivision schemes, since new control points are a weighted average of the old control points. The de Rham-Chaikin mask is given by

$$a_j = \frac{1}{4}\begin{pmatrix} 3 \\ j \end{pmatrix}, \quad j = 0, \ldots, 3; \tag{22}$$

the corner-cutting property of this mask is illustrated in Figure 5.



(a) control points: original (•), after one iteration (∘) *(b) original control points (•), control polygon after 6 iterations (–)*

Figure 5: Subdivision with the de Rham-Chaikin mask Equation 22

Subdivision with the de Rham-Chaikin mask is convergent [9, 11], so that we are guaranteed the existence of a refinable function. This refinable function turns out to be the B-spline of degree 2 (order 3), denoted by B_2. Accordingly, the limit curve of this subdivision scheme is the quadratic spline

$$f(x) = \sum_j c_j^{(0)} B_2(x-j), \quad x \in \mathbb{R}. \tag{23}$$

The generalization of this scheme is known as the Lane-Riesenfeld scheme of order m. The Lane-Riesenfeld scheme of order m has mask

$$a_j^{(m)} = \frac{1}{2^{m-1}} \left(\begin{array}{c} m \\ j \end{array} \right), \quad j = 0, \ldots, m \tag{24}$$

and its limit curve is the spline of degree $m-1$ (see [10]) defined by

$$f(x) = \sum_j c_j^{(0)} B_{m-1}(x-j) \tag{25}$$

where B_{m-1} is the B-spline of degree $m-1$. Note that the smoothness of the limiting curve increases with m.

Note also that the Lane-Riesenfeld mask has finite support, i.e.

$$a_j^{(m)} = 0, \quad j \notin [0, m], \tag{26}$$

and that the mask elements within the support are all positive, i.e.

$$a_j^{(m)} > 0, \quad j \in [0, m]. \tag{27}$$

General results for finitely supported positive masks can be found in [12, 1].

All that remains to be done in the Lane-Riesenfeld example of corner-cutting subdivision is to modify the scheme in the presence of boundaries. The problem is the same as before—we need to supply missing values at the boundary. A very simple procedure is to repeat the boundary values as many times as needed. It turns out that this again leads to a a set of refinable functions, this time splines with multiple knots at the boundary. Thus the boundary-modified scheme again converges to a spline of the same degree as defined by the interior mask.

An example of these modified refinable functions is shown in Figure 6 and of the boundary-modified subdivision

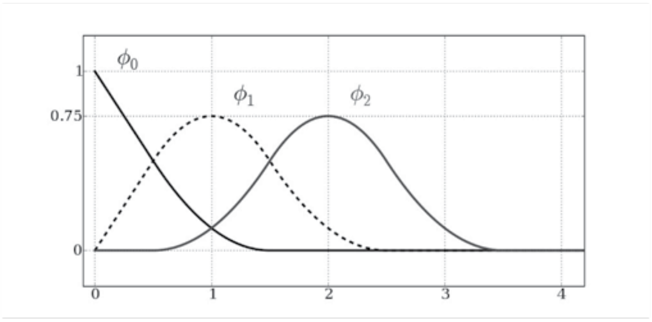
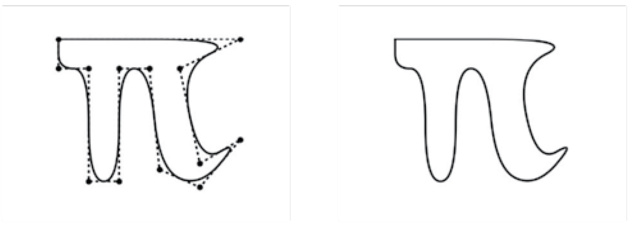


Figure 6: Boundary modified de Rham-Chaikin refinable functions

in Figure 7. Note we have chosen the top left hand control point as the first and last control points, hence we get a sharp corner at this point.



(a) original control points (•) and control polygon after 6 iterations (–) *(b) control polygon after 6 iterations*

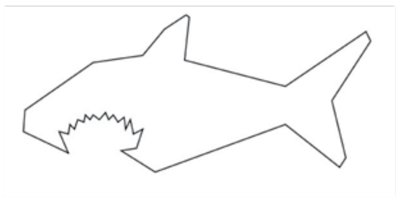
Figure 7: Boundary modified de Rham-Chaikin subdivision

Next consider the control polygon of a shark shown in Figure 8(a). Here we applied the standard de Rham-Chaikin, iterated to convergence, and obtained the sorry-looking shark of Figure 8(b)—a shark with blunt teeth is no shark at all. Doubling the control points defining the teeth results in the much happier-looking shark of Figure 8(c).

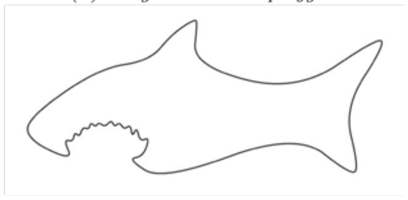
4. EXAMPLES

In this section we apply interpolatory and corner-cutting subdivision schemes to a few practical problems. The first example is a dynamic signature obtained via a digitising tablet. Figure 9(a) shows the original signature and the discretisation effects of this particular tablet should be obvious.

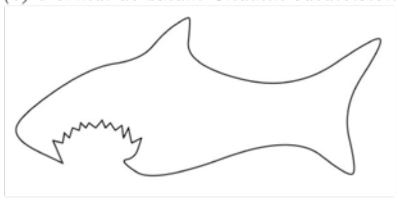
Since the samples are connected by straight lines, Fig-



(a) Original control polygon



(b) Normal de Rham-Chaikin subdivision



(c) de Rham-Chaikin subdivision with double points defining the teeth

Figure 8: Keeping corners by doubling control points

ure 9(a) can also be viewed as an example of a Dubuc-Deslauriers subdivision scheme with $n = 1$. Figure 9(b) shows the result of applying successive Dubuc-Deslauriers with $n = 2$ and downsampling. It is clearly smoother than Figure 9(a) but not as smooth as the de Rham-Chaikin corner-cutting subdivision and downsampled curve shown in Figure 9(c).

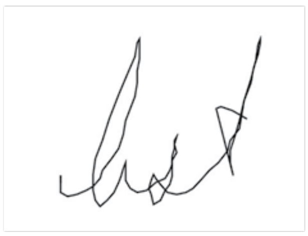
The next example increases the resolution of an image through subdivision. (Do not confuse this interpolation procedure with super resolution techniques where the resolution is increased by extracting additional information from multiple images.) Figure 10(a) shows the original image with a subsampled version shown in Figure 10(b).

We now apply subdivision to the subsampled version in an effort to recover the original. Figure 10(c) and (d) show the results of using the interpolatory Dubuc-Deslauriers scheme with $n = 2$ and the corner-cutting de Rham-Chaikin subdivision schemes, respectively. It is left to the reader to decide which one of the two schemes provide the more acceptable results.

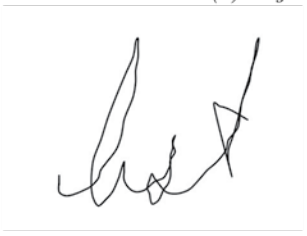
5. SURFACE SUBDIVISION

In the preceding sections we have limited our discussion to subdivision of curves. In this section we present the Doo-Sabin subdivision scheme [13] as an example of surface subdivision. Examples of other subdivision schemes are described in Catmull and Clark [14], and Loop [15]. For a good introduction see e.g. [16].

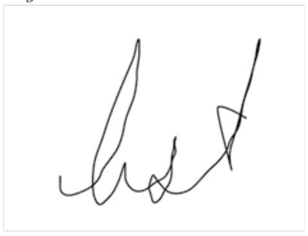
The Doo-Sabin scheme is based on uniform quadrilat-



(a) original Signature

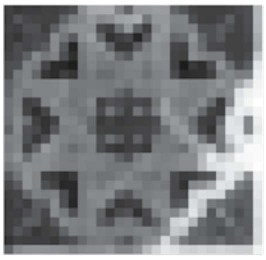


(b) interpolatory subdivision

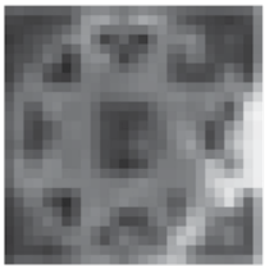


(c) corner-cutting subdivision

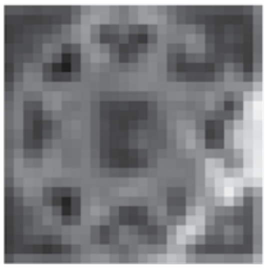
Figure 9: Smoothing a signature



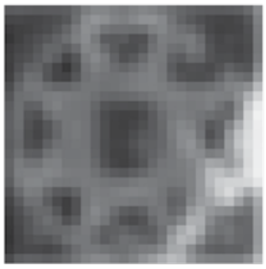
(a) original image



(b) subsampled image



(b) interpolatory subdivision



(c) corner-cutting subdivision

Figure 10: Smoothing an image (detail)

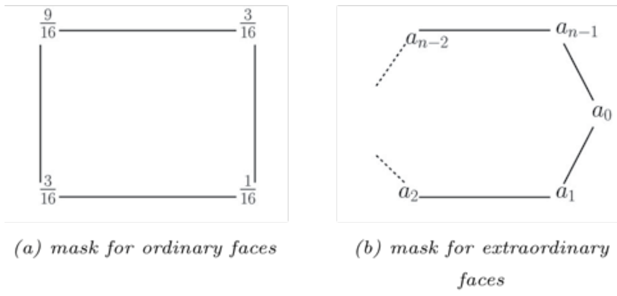
eral faces. It is a vertex split method that is based on biquadratic B-spline subdivision.

This scheme uses only one mask for all quadrilateral faces, shown in Figure 11(a). This mask is fitted to each face with the weights used cyclically to result in four children vertices per quadrilateral face.

Since some faces in the mesh will not be quadrilateral this mask will not always fit the faces. For the extraordinary faces (non-quadrilateral faces) we use a variable mask: for a face with n vertices we use the mask in Figure 11(b) with

$$a_i = \frac{1}{4n}\begin{cases} n+5, & i=0, \\ 3+2\cos(\frac{2i\pi}{n}), & i \in \{1,\ldots,n-1\}. \end{cases} \tag{28}$$

Notice that the mask for the extraordinary faces re-

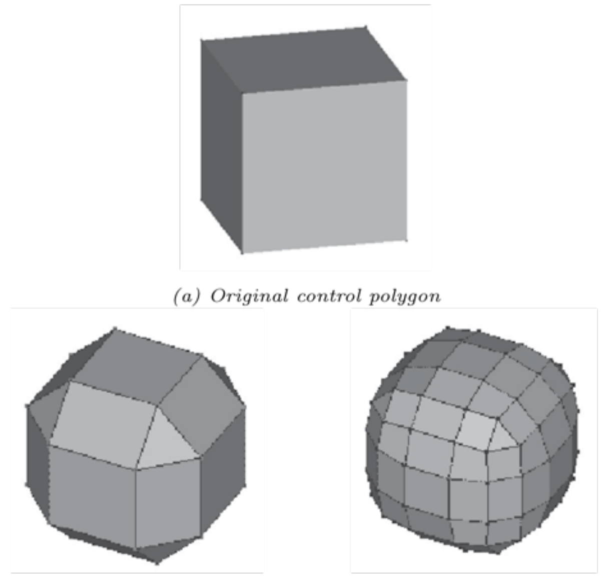


(a) mask for ordinary faces *(b) mask for extraordinary faces*

Figure 11: Doo-Sabin Subdivision scheme masks

duces to the mask for an ordinary face when $n = 4$.

In Figure 12, the first two iterations of this subdivision scheme are shown when applied to a unit cube.



(a) Original control polygon

(b) Mesh after first iteration *(c) Mesh after second iteration*

Figure 12: Doo-Sabin subdivision of a cube

Once again the nature of the subdivision scheme and the limiting surface depends entirely on the choice of mask. For surface subdivision using triangular meshes, see [15].

6. CONCLUSION

In this paper a brief overview of subdivision schemes for curves was given. The main ideas were explained for curves. In particular, interpolatory and corner-cutting schemes were discussed and the necessary boundary modifications for finite sequences were derived. The generalization to surfaces was briefly discussed. The numerical examples demonstrate the power of these methods to generate smooth curves and surfaces from a limited number of control points.

7. REFERENCES

[1] C. A. Micchelli, *Mathematical Aspects of Geometric Modeling.* CBMS-NSF Regional Conference Series in Applied Mathematics, SIAM, Philadelphia, PA, 1995.

[2] S. Dubuc, “Interpolation through an iterative scheme,” *J. Math. Anal. Appl.*, vol. 114, pp. 185–204, 1986.

[3] G. Deslauriers and S. Dubuc, “Symmetric iterative interpolation processes,” *Constr. Approx.*, vol. 5, no. 1, pp. 49–68, 1989.

[4] D. Knuth, *Digital Typography.* CSLI Publications, Stanford, California, 1999.

[5] C. A. Micchelli, “Interpolatory subdivision schemes and wavelets,” *J. Approx. Theory*, vol. 86, pp. 41–71, 1996.

[6] M. Neamtu, “Convergence of subdivision versus solvability of refinement equations,” *East Journal of Approximations*, vol. 5, no. 2, pp. 183–210, 1999.

[7] D. L. Donoho, “Smooth wavelet decompositions with blocky coefficient kernels,” in *Recent Advances in Wavelet Analysis* (L. L. Schumaker and G. Webb, eds.), vol. 3 of *Wavelet Analysis and Its Applications*, pp. 259–308, Academic Press, Boston, 1994.

[8] J. M. de Villiers, K. M. Goosen, and B. M. Herbst, “Dubuc–Deslauriers subdivision for finite sequences and interpolation wavelets on an interval,” *SIAM J. Math. Anal.*, vol. 35, no. 2, pp. 423–452, 2003.

[9] G. de Rham, “Sur une courbe plane,” *Journal de Mathématiques Pures et Appliquées*, vol. 35, pp. 25–42, 1956.

[10] J. M. Lane and R. F. Riesenfeld, “A theoretical development for the computer generation and display of piecewise polynomial surfaces,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 2, pp. 35–46, 1980.

[11] G. M. Chaikin, “An algorithm for high-speed curve generation,” *Computer Graphics and Image Processing*, vol. 3, pp. 346–349, 1974.

[12] C. A. Micchelli and H. Prautzsch, “Refinement and subdivision for spaces of integer translates of a compactly supported function,” in *Numerical Analysis* (D. F. Griffiths and G. A. Watson, eds.), pp. 192–222, Longman, London, 1987.

[13] D. Doo and M. Sabin, “Behaviour of recursive division surfaces near extraordinary points,” *Computer-Aided Design*, vol. 10, pp. 356–360, September 1978.

[14] E. Catmull and J. Clark, “Recursively generated b-spline surfaces on arbitrary topological meshes,” *Computer-Aided Design*, vol. 10, pp. 350–355, September 1978.

[15] C. Loop, “Smooth subdivision surfaces based on triangles,” Master’s thesis, University of Utah, Department of Mathematics, 1987.

[16] J. O. M. Larsen and C. K. Madsen, “Subdivision surfaces,” tech. rep., Department of Mathematics, Technical University of Denmark, July 2001.

UNSUPERVISED CLASSIFICATION OF DYNAMIC FROTHS.

G. Forbes* and G. de Jager**

** Centre for Minerals Research, Dept. of Chemical Engineering, University of Cape Town, Private Bag, Rondebosch, 7701, South Africa*
*** Digital Image Processing Group, Dept. of Electrical Engineering, University of Cape Town, Private Bag, Rondebosch, 7701, South Africa*

Abstract: Machine vision systems typically classify images of a flotation froth surface into one of a distinct set of classes. This process typically involves having an experienced operator identify a set of froth classes. After this, a machine vision system is trained to identify these froth classes. Identifying these froth classes is particularly challenging for froths which have "dynamic" bubble size distributions. Using unsupervised clustering algorithms, it is possible to automatically learn these froth classes without user input. Validation of this technique is done by showing that the identified froth classes have statistically different relationships between the froth velocity and concentrate grade.

Key words: machine vision, froth flotation, unsupervised classification, bubble size distribution.

1. INTRODUCTION

1.1 Flotation

Flotation is a separation process used in many mining operations to upgrade the desired mineral concentration before further downstream processing. The operation of the flotation process is a complex one which is not entirely understood. Each flotation cell has numerous input parameters (reagent dosage, froth depth, air flow rate) and is also affected by numerous disturbance variables (ore type, mill performance). Typically, plant operators inspect the state of the froth visually, taking into account such parameters as velocity, bubble size, texture, colour and stability. Based on the state of the froth, the operator might make changes to one or more of the input parameters in order to achieve optimal performance.

As a result of this, numerous machine vision systems have been developed to analyse the state of the froth in a manner similar to that of an experienced plant operator. The advantage of such an instrument is the availability of precise, unbiased measurements 24 hours a day.

1.2 Froth Classification

Machine vision systems that monitor froth flotation cells typically classify the state of the froth into a number of discreet classes [1, 2, 3]. These froth classes are usually dependent on the bubble size distribution (texture) of the froth. The reason for using both bubble size and texture measurements is that it is not always possible to accurately segment individual bubbles in a froth. This means that it is not always possible to determine an accurate bubble size distribution (BSD) for a froth. Under such circumstances, texture measures can be used which allow for the discrimination of froths with different bubble size distributions. However, the texture measures do not provide the user with an accurate bubble size distribution.

The froth classes are usually determined by studying the fluctuations in a flotation cell over a long period of time (typically a number of days). Visually dissimilar froth classes are then identified by experienced operators. The machine vision system is then trained to be able to identify whether or not the cell being monitored is in one of these predetermined froth classes. One of the disadvantages of such a method is that there is no guarantee that all possible froth classes will be identified during the training process. This means that the system will not be able to provide useful information when an unknown froth class is identified.

1.3 "Dynamic" Bubble Size Distributions

When the bubble size distribution of the froth being monitored does not change rapidly over a short period of time, identification of froth classes by an experienced operator is a relatively simple process. This is not always the case; froths exist which have "dynamic" bubble size distributions. An example of such a froth is shown in Figure 1, where two frames of video footage which have been taken within one second of each other are shown.

This paper will be dealing with froths which have these "dynamic" bubble size distributions. One of the biggest difficulties with these froths is identifying different froth classes. This is because two different froths will look similar at times when viewed side by side. This dynamic nature of the froth makes it very difficult to classify froths into the appropriate classes. It is also a very time consuming task, that is likely to have multiple operators classifying the same data set into different resultant subsets. Having a solution to finding these froth classes with minimal operator intervention is of utmost importance as it results in the availability of consistent results within a reasonable time frame.

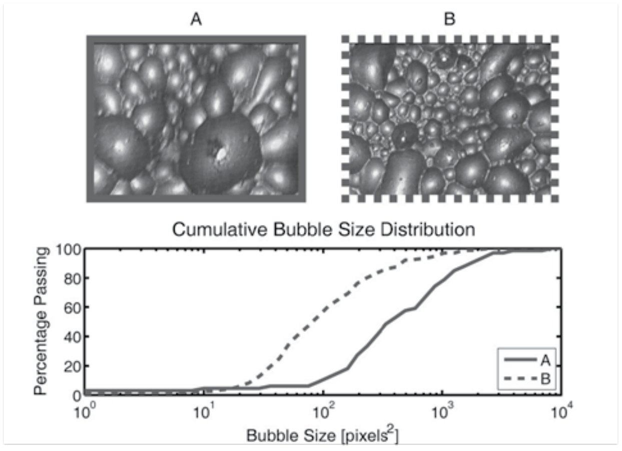


Figure 1: An example of a froth with a "dynamic" BSD. The BSDs are taken from two frames of video which are within 1 second of each other.

1.4 Objectives

The specific objectives of this paper are to show that unsupervised classification algorithms can be used to automatically detect a user specified number of froth classes. It will also be shown that the froth classes identified are not random groupings of froth classes, but are in fact real froth classes that are statistically significantly different in terms of the metallurgical performance of the cell at the time of operation when the froth class was observed.

2. UNSUPERVISED CLASSIFICATION OF FROTHS

2.1 Froth Data Set

The data set used in this work consists of 105 video segments. Each of these video segments is one minute in duration (1500 frames) and was captured from the first cell of the copper rougher circuit at Kennecott Utah Copper Concentrator in January 2006. At the same time that the video footage was captured, metallurgical samples of the feed, concentrate and tailings of the cell being monitored were taken. These samples were later analysed to determine their elemental composition.

Some example images of the froth video segments collected are shown in Figure 2. It is important to note that the still images do not capture the dynamic nature of the froths.

2.2 Bubble Size Distribution Measurements

The bubble size distribution for each of the frames of video from all 105 videos was calculated using the improved watershed segmentation technique [4]. As has been shown previously, further reduction of the data in the bubble size distribution to a mean, median or p80 value is not appropriate for dynamic froths such as the one being examined here [5].

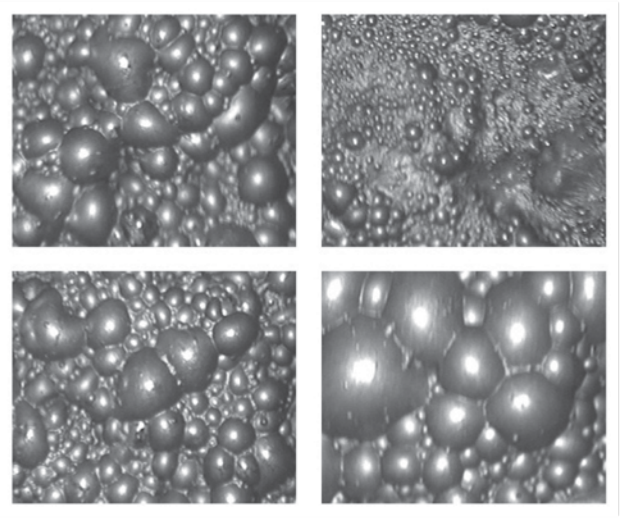


Figure 2: Example images of the froth of the first rougher cell at Kennecott. Note the different bubble sizes. Single still images do not provide an accurate description of the "dynamic" nature of the froth.

2.3 Frequently Occurring Bubble Size Distributions

The frequently occurring BSD algorithm is inspired by the work of Varma and Zisserman [6], but instead of finding image *textons* that occur frequently over an image, *frequently occurring BSDs* are determined that are found from the bubble size data.

A random sample of frames is taken from the entire data set of froth videos, such that the sample is representative of all the different bubble size distributions that can be found in the data set. The cumulative bubble size distributions are then calculated for each of the frames in the sample. Once this has been done, an unsupervised furthest-neighbour clustering algorithm is used to split the sample into a user specified number of classes. This is achieved by firstly creating a intra-distance matrix for the entire set of cumulative BSD samples. The Kolmogorov-Smirnov distance measure [7] is used to calculate the distance between two cumulative BSDs. The Matlab statistics toolbox is used to perform the unsupervised furthest-neighbour clustering. The intra-distance matrix is passed to the linkage function, the output of which is passed to the cluster function. This generates the user specified number of classes. For this work, eight clusters are typically used. This value is chosen so that the number of classes is small enough to ensure that there is still a visual difference between the images from which the identified BSDs are generated. The mean cumulative bubble size distribution can then be calculated for each of

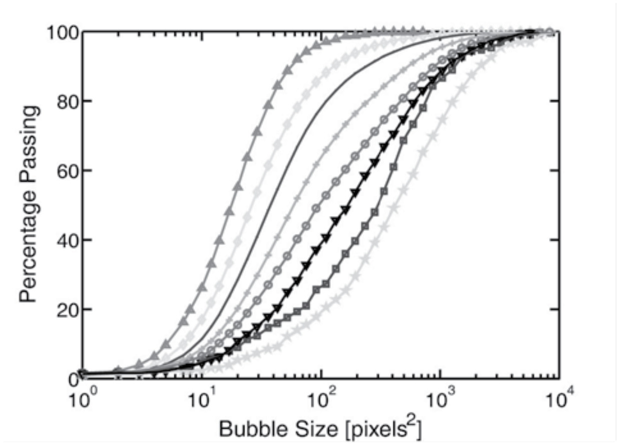


Figure 3: Frequently occurring BSDs learnt from 500 samples.

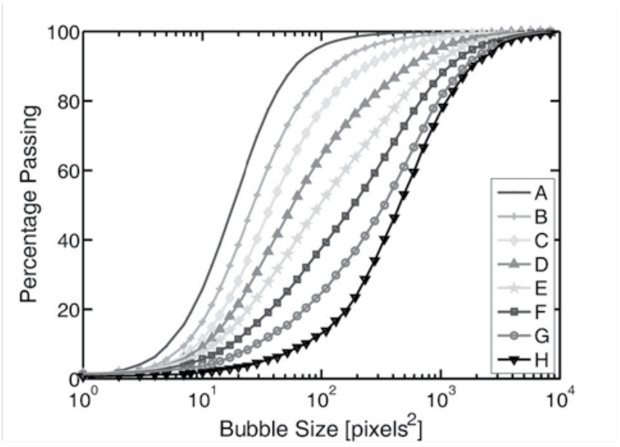


Figure 4: Frequently occurring BSDs learnt from 9000 samples.

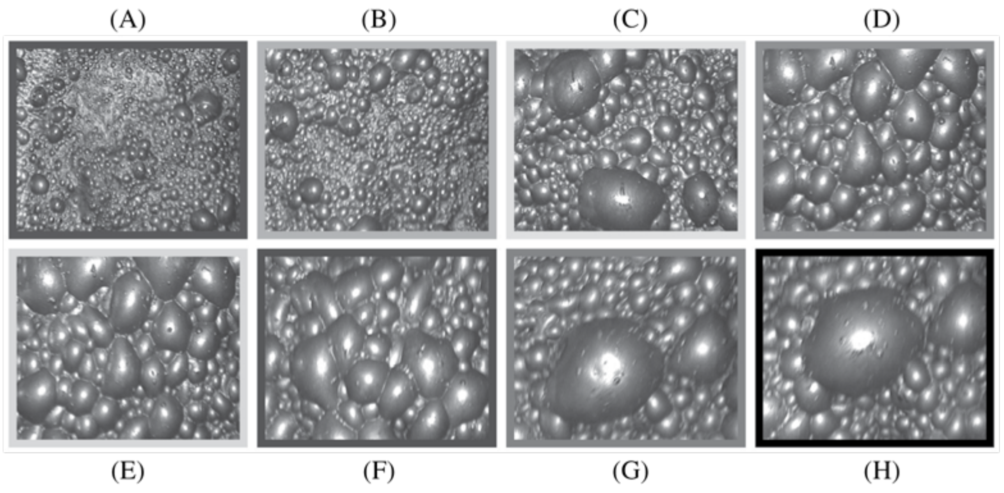


Figure 5: Example images of the froths represented by the cumulative BSDs in Figure 4.

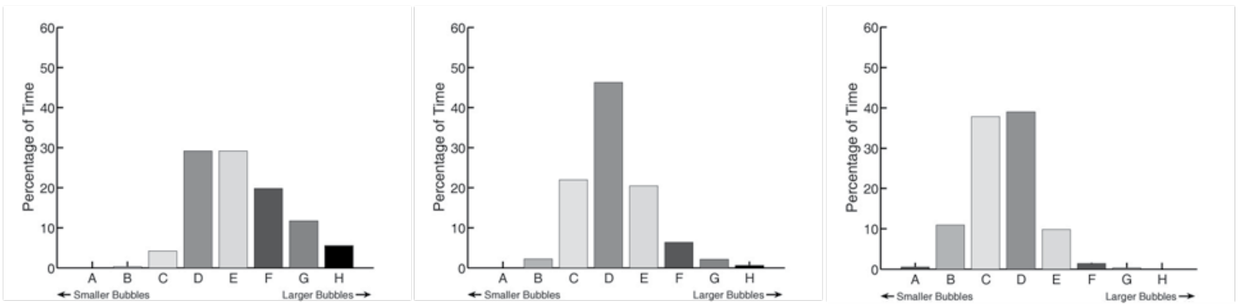


Figure 6: The results of using unsupervised classification to determine three froth classes. The labels of the histograms correspond to the labels of the BSDs in Figure 4.

segments. This resulting set of cumulative bubble size distributions is known as the frequently occurring BSDs [8].

To ensure that the random sample of frames taken from the entire data set was representative, a test was performed by looking at the resultant frequently occurring BSDs that were found for different numbers of samples. The results from these tests are shown in Figures 3 and 4. These figures show the resulting frequently occurring BSDs when the number of samples drawn are 500 and 9000 respectively. From these figures

Figures 3 and 4. These figures show the resulting frequently occurring BSDs when the number of samples drawn are 500 and 9000 respectively. From these figures it is evident that a vast increase in the size of sample used in the unsupervised clustering does not have a significant impact on the resultant frequently occurring BSDs. The only difference is that the smoothness of the BSDs is increased when larger sample sizes are used. The difference in labelling of the frequently occurring BSDs is a result of the order in which the outputs from the clustering algorithm are generated, so the differences in labelling can be ignored.

Figure 5 shows example images of froths corresponding to the frequently occurring BSDs shown in Figure 4.

2.4 Characterisation of Dynamic Froths

Using the frequently occurring cumulative bubble size distributions, it is possible to characterise each video segment as a histogram. The histogram has the same number of bins as the number of frequently occurring BSDs that were identified in Section 2.3. The histogram shows the percentage of time that the froth has a bubble size distribution similar to the frequently occurring bubble size distributions.

The chi-squared distance measure [9] can be used to provide a measure of dissimilarity between the characteristic histograms of different froths. It is possible to create a dissimilarity matrix for the entire data set of characteristic histograms of froth video segments. This can be used in an unsupervised clustering algorithm (furthest-neighbour) to classify the data set into classes with similar characteristic histograms. Once again, the Matlab statistical toolbox functions: linkage and cluster are used to do the clustering.

The results from using these unsupervised clustering algorithms are shown in Figure 6. Note that the labels of the bars in Figure 6 corresponds to the bubble size distributions with the same labels in Figures 4 and 5. The characteristic histograms are clustered into three froth classes. This number is chosen for two reasons: firstly, experience tells us that flotation cells typically have between three to five different froth classes under normal operating conditions and secondly, to maximise the amount of concentrate data per froth class, which is important to ensure that statistically meaningful results are obtained.

3. VALIDATION

Froth velocity is an important performance indicator which is typically used for mass pull and concentrate grade prediction. In this section, the link between froth velocity and concentrate grade is used to validate the froth classes identified by the clustering algorithm described in Section 2.

3.1 Metallurgical Responses of Froth Classes

For each of the identified froth classes, the relationship between the froth velocity and metallurgical content of the concentrate can be modelled by linear regression. Different froth classes will have different trends, and so it is possible to use this information to determine if the froth classes identified are correct or just random collections of froths.

If the froth classes identified by the unsupervised classification algorithm are no more than a random selection of froths, then the relationships between the froth velocity and concentrate metallurgy for each of the froth classes will not be statistically significantly different from each other.

An example of such a set of regression lines is shown in Figure 7 which corresponds to a set of three froth classes which have been generated by randomly selecting their membership. The values in Table I show the results from an analysis to determine whether or not the regression lines from each of the randomly created froth classes are statistically different. The values are all less than ninety-five percent. This indicates that one cannot say with confidence that the lines are statistically different, and must therefore accept the null hypothesis which is that there is no difference between these froth classes. This is exactly what is to be expected from randomly allocated froth classes.

3.2 Statistical Calculations

This section gives a brief overview of the statistical tests used for the comparison of regression lines from different froth classes. For more detail, the reader is referred to [9,10]. The following series of F-Tests are performed in order to determine whether or not the regression lines are statistically different:

1. F-Test for the comparison of data sets' variance
2. F-Test for the comparison of the slopes of the regression lines

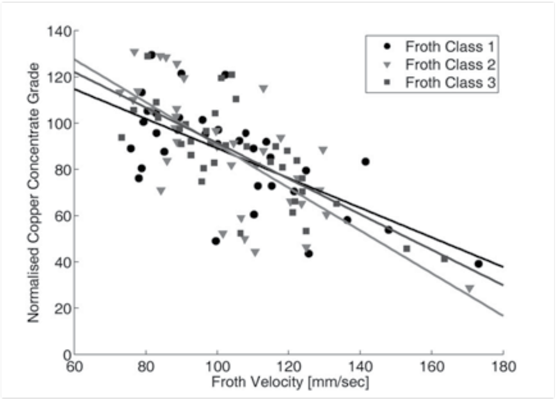


Figure 7: Linear trends relating froth velocity to concentrate grade for three randomly allocated froth classes.

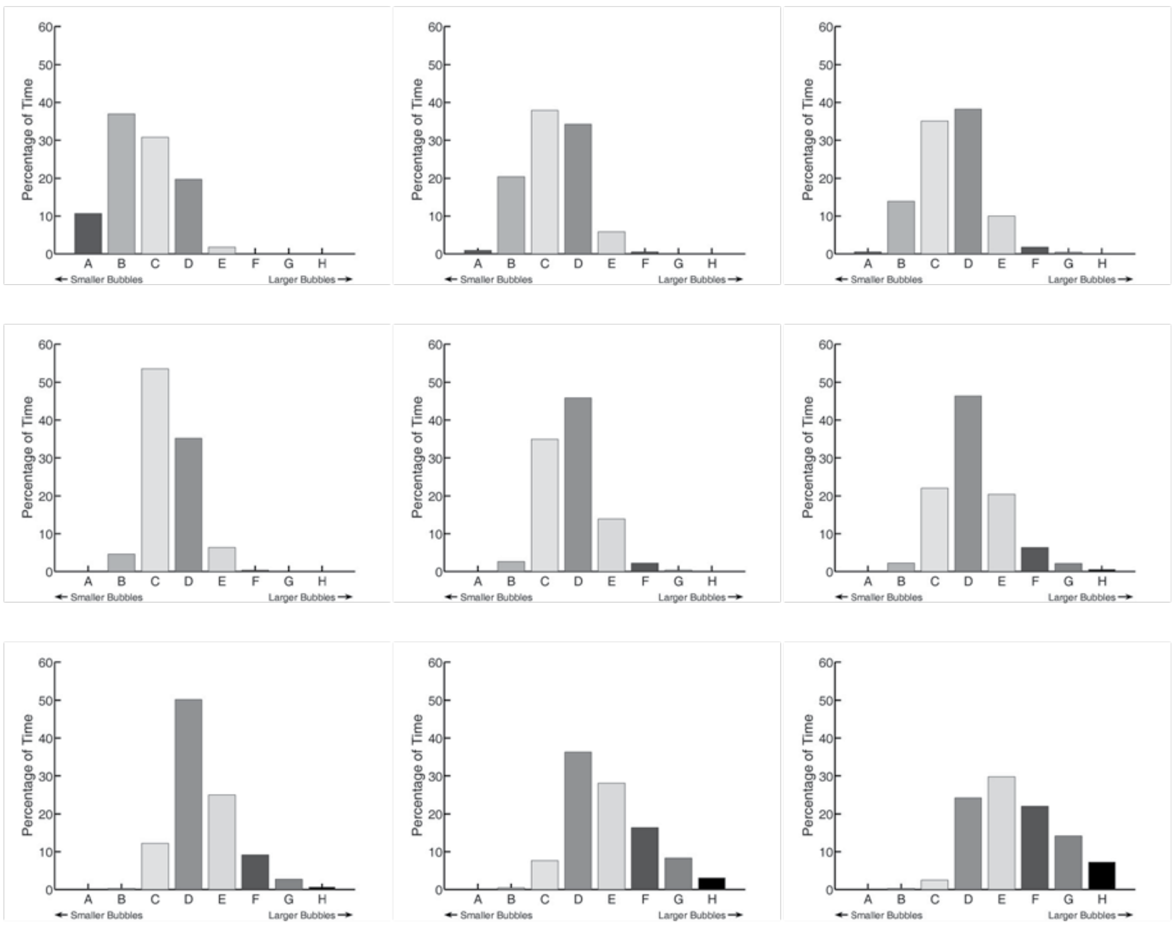


Figure 8: The results of using unsupervised classification to determine nine froth classes. The labels of the histograms correspond to the labels of the BSDs in Figure 4.

Table I: Summary of comparative statistics for three randomly allocated froth classes.

Assay	Froth Class A	Froth Class B	Confidence of Difference in Slope	Confidence of Difference in Intercept	Confidence of Difference in Mean	Confidence of Difference in (Overall)
Copper [Cu]	1	2	84.08	10.37	55.23	84.08
	1	3	53.04	34.01	66.95	66.95
	2	3	56.61	29.64	64.78	66.78

Table II: Summary of comparative statistics for three froth classes.

Assay	Froth Class A	Froth Class B	Confidence of Difference in Slope	Confidence of Difference in Intercept	Confidence of Difference in Mean	Confidence of Difference in (Overall)
Copper [Cu]	1	2	7.89	99.91	99.95	99.95
	1	3	5.11	99.99	100.00	100.00
	2	3	61.40	99.91	99.96	99.96

Table III: Summary of comparative statistics three froth classes generated using the alternative approach.

Assay	Froth Class A	Froth Class B	Confidence of Difference in Slope	Confidence of Difference in Intercept	Confidence of Difference in Mean	Confidence of Difference in (Overall)
Copper [Cu]	1	2	63.07	99.99	100.00	100.00
	1	3	95.30	100.00	100.00	100.00
	2	3	94.58	100.00	100.00	100.00

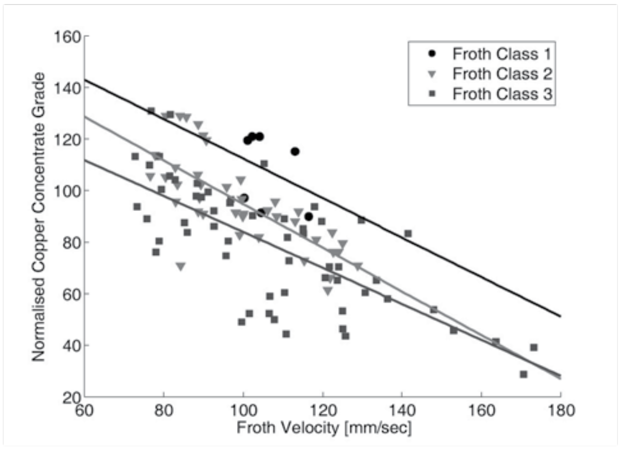


Figure 9: Linear trends relating froth velocity to concentrate grade for three froth classes.

3. F-Test for the comparison of the intercepts of the regression lines
4. F-Test for the comparison of the mean of the regression lines.

3.3 Verification of Unsupervised Clustering

Figure 9 shows the resulting trends from using the fitting of a linear regression model relating froth velocity to normalised concentrate grade for each of the three froth classes determined by the unsupervised clustering algorithm (the copper concentrate grade has been normalised for confidentiality reasons. The normalisation process does not however affect the trends observed.) It is clear from the figure that the trends have different mean values, unlike the random allocation of froths in Figure 7.

Statistical analysis of the differences between these regression lines is show in Table II. All three of the regression lines are different from each other with at least 99.95% confidence.

These results show that the techniques used here to automatically determine the froth classes present in a set of videos of dynamic froths give meaningful results and not just a random selection of froth classes.

4. AN ALTERNATIVE APPROACH

4.1 User Intervention

An alternative approach to the previously mentioned classification method is now discussed. Unlike the previous technique, which only relies on the user's input for the number of froth classes to be determined, this method makes use of user intervention in deciding how to merge a larger set of froth classes into a smaller, more manageable set of froth classes.

Once again, the unsupervised froth classification techniques described earlier were used to determine nine froth classes. These nine froth classes are shown in

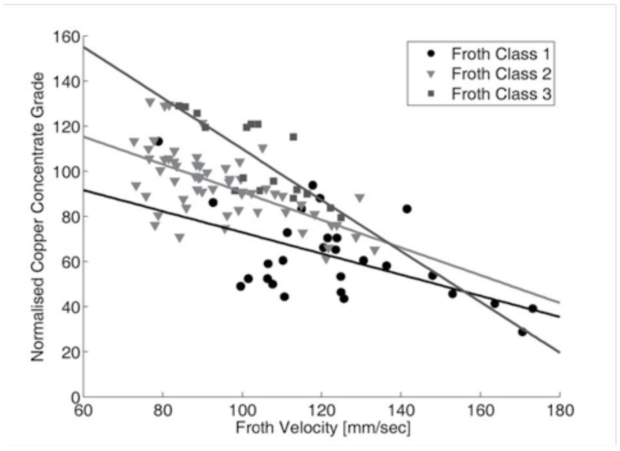


Figure 10: Linear trends relating froth velocity to concentrate grade for three froth classes using the alternative approach.

Figure 8. The metallurgical (froth velocity vs. concentrate grade) responses of these classes were analysed, and froths which froths were grouped together. It is necessary to do this, as having nine froth classes results in the metallurgical data effectively being divided by nine, thus reducing the statistical confidence that can be put on the observed trends. The result of this grouping of froth classes resulted in the middle column of froth classes in Figure 8 being chosen as the final froth classes.

The final metallurgical responses of these froth classes is shown in Figure 10. It is clear that different linear regression models exist for the separate froth classes. This is confirmed in Table III which shows the results for testing the statistical significance of the regression lines being the same. All of the trends relating froth velocity to concentrate grade for these froth classes have a 99.9% confidence that they are statistically different. It is also interesting to note that for this set of froth classes, the trends have an almost 95% confidence that the slopes are statistically different. This information is particularly useful from an operational point of view, as it provides the operator with additional information which can be used for the improvement of operation of the cell being monitored.

5. CONCLUSION

It has been shown that unsupervised clustering techniques can be used to separate a set of dynamic froths into visually similar froth classes. The results from the clustering have been validated by showing that each of the froth classes identified have statistically different regression lines relating froth velocity to concentrate grade. This would not be the case if the froth video segments were divided into random classes.

An alternative approach which relies more on user intervention results in very similar froth classes being identified. This is a further indication that the technique is giving appropriate clusterings of data. The alternative approach also provides additional information for the operators as the trends modelled have statistically different slopes relating froth velocity to concentrate grade.

The major advantages of having a system which is able to make use of unsupervised clustering are the consistency of the results, the speed at which they can be obtained and the fact that this method will enable new froth classes to be identified in an online setting. This is unlike manually identifying froth classes which is a difficult, time consuming and operator dependent process, which invariably results in a poor set of froth classes to work with.

ACKNOWLEDGEMENTS

The authors would like to thank the following for their financial support: the NRF, the Department of Labour, Rio Tinto, Anglo Platinum and UCT Department of Chemical Engineering.

6. REFERENCES

[1] G. Bartolacci, P. Pelletier, J. Tessier, C. Duchesne, P.A. Boss, and J. Fournier, "Application of numerical image analysis to process diagnosis and physical parameter measurement in mineral processes part i: Flotation control based on froth textural characteristics," *Centenary of Flotation Symposium*, Brisbane, Queensland, pp. 73 – 84, 2005.

[2] A. Cipriano, M. Guarini, R. Vidal, A. Soto, C. Seplveda, D. Mery, and H. Briseo, "A real time visual sensor for supervision of flotation cells," *Minerals Engineering*, vol. 11, pp. 489 – 499, 1998.

[3] M. Lu, W. Liu, F. Wang, and Y. Wang, "Using image analysis method to evaluate the performance of coal flotation for industrial column," *19th International Pittsburgh Coal Conference*, Pittsburgh, 2002.

[4] G. Forbes and G. de Jager, "Texture measures for improved watershed segmentation of froth images," in Fifteenth Annual Symposium of the Pattern Recognition Association of South Africa, Grabouw, pp. 1–6, 2004.

[5] G. Forbes and G. de Jager, "Bubble size distributions for froth classification," *Sixteenth Annual Symposium of the Pattern Recognition Association of South Africa*, Langebaan, South Africa, pp. 99–104, 2005.

[6] M. Varma and A. Zisserman, "A statistical approach to texture classification from single images," *International Journal of Computer Vision*, vol. 62, no. 1–2, pp. 61–81, 2005.

[7] Y. Rubner, J. Puzicha, C. Tomasi, and J. M. Buhmann, "Empirical evaluation of dissimilarity measures for color and texture," *Computer Vision and Image Understanding Journal*, vol. 84, no. 1, pp. 25–43, 2001. [8] G R Forbes, G de Jager, and D J Bradshaw, "Effective use of bubble size distribution measurements," *XXIII International Mineral Processing Congress*, Istanbul, Turkey, pp. 554–559, 2006.

[9] T. J. Napier-Munn, "An introduction to comparative statistics and experimental design for minerals engineers," 2005.

[10] T. J. Napier-Munn, "Analysing plant trials by comparing recovery-grade regression lines," *Minerals Engineering*, vol. 11, no. 10, pp. 949 – 958, 1998.

FUZZY ARTMAP AND NEURAL NETWORK APPROACH TO ONLINE PROCESSING OF INPUTS WITH MISSING VALUES

F.V. Nelwamondo and T. Marwala

School of Electrical and Information Engineering, University of the Witwatersrand, Johannesburg, Private Bag 3, Wits, 2050, South Africa.

Abstract: An ensemble based approach for dealing with missing data, without predicting or imputing the missing values is proposed. This technique is suitable for online operations of neural networks and as a result, is used for online condition monitoring. The proposed technique is tested in both classification and regression problems. An ensemble of Fuzzy-ARTMAPs is used for classification whereas an ensemble of multi-layer perceptrons is used for the regression problem. Results obtained using this ensemble-based technique are compared to those obtained using a combination of auto-associative neural networks and genetic algorithms and findings show that this method can perform up to 9% better in regression problems. Another advantage of the proposed technique is that it eliminates the need for finding the best estimate of the data, and hence, saves time.

Key words: Autoencoder neural networks, Fuzzy-ARTMAP, Genetic algorithms, Missing data, Multi-layer Perceptron MLP

1. INTRODUCTION

Real time processing applications that are highly dependent on the newly arriving data often suffer from the problem of missing data. In cases where decisions have to be made using computational intelligence techniques, missing data become a hindering factor. The biggest challenge on one hand is that most computational intelligence techniques such as neural networks are not able to process input data with missing values and hence, cannot perform classification or regression when some input data are missing. Various heuristics for missing data have however been proposed in the literature [1]. The simplest method is known as 'listwise deletion' and this method simply deletes instances with missing values [1]. The major disadvantage of this method is the dramatic loss of information in data sets. There is also a well documented evidence showing that ignorance and deletion of cases with missing entries is not an effective strategy [1-2]. Other common techniques are imputation methods based on statistical procedures such as mean computation, imputing the most dominant variable in the database, hot deck imputation and many more. Some of the best imputation techniques include the Expectation Maximization (EM) algorithm [3] as well as neural networks coupled with optimisation algorithms such as genetic algorithms as used in [4] and [5]. Imputation techniques where missing data are replaced by estimates are increasingly becoming popular. A great deal of research has been done to find more accurate ways of approximating these estimates. Among others, Abdella and Marwala [4] used neural networks together with Genetic Algorithms (GA) to approximate missing data. Gabrys [6] has also used Neuro-fuzzy techniques in the presence of missing data for pattern recognition problems.

The other challenge in this work is that, online condition monitoring uses time series data and there is often a limited time between the readings depending on how frequently the sensor is sampled. In classification and regression tasks, all decisions concerning how to proceed must be taken during this finite time period. Methods using optimisation techniques may take longer periods to converge to a reliable estimate and this depends entirely on the complexity of the objective function being optimised. This calls for better techniques to deal with this missing data problem.

We argue in this paper that it is not always necessary to have the actual missing data predicted. Differently said, it is not in all cases that the decision is dependent on *all* actual values. Therefore, a vast amount of computational resources is wasted in attempts to predict the missing values, whereas the ultimate result could have been achieved without such values. In light of this challenge, this paper investigates a problem of condition monitoring where computational intelligence techniques are used to classify and regress in the presence of missing data without the actual prediction of missing values. A novel approach where no attempt is made to recover the missing values, for both regression and classification problems, is presented. An ensemble of fuzzy-ARTMAP classifiers to classify in the presence of missing data is proposed. The algorithm is further extended to a regression application where Multi-layer Perceptron (MLP) is used in an attempt to get the correct output with limited input variables. The proposed method is compared to a technique that combines neural networks with Genetic Algorithm (GA) to approximate the missing data.

2. MISSING DATA THEORY

According to Little and Rubin [1], missing data are categorized into three basic types namely: 'Missing at Random', (MAR), 'Missing Completely at Random', (MCAR) and 'Missing Not at Random', (MNAR). MAR is also known as the ignorable case [3]. The probability of datum d from a sensor S to be missing at random is dependent on other measured variables from other sensors. A simple example of MAR is when sensor T is only read if sensor S reading is above a certain threshold. In this case, if the value read from sensor S is below the threshold, there will be no need to read sensor T and hence, readings from T will be declared missing at random. MCAR on the other hand refers to a condition where the probability of S values missing is independent of any observed data. In this regard, the missing value is neither dependent on the previous state of the sensor nor any reading from any other sensor. Lastly, MNAR occurs when data is neither MAR nor MCAR and is also referred to as the non-ignorable case [1, 3] as the missing observation is dependent on the outcome of interest. A detailed description of missing data theory can be found in [3]. In this paper, we shall assume that data is MAR.

3. BACKGROUND

3.1 Neural network: multi-layer perceptrons

Neural networks may be viewed as systems that learn the complex input-output relationship from any given data. The training process of neural networks involves presenting the network with inputs and corresponding outputs and this process is termed *supervised learning*. There are various types of neural networks but we shall only discuss the MLP since they are used in this study. MLPs are feed-forward neural networks with an architecture comprising of the input layer, hidden layer and the output layer. Each layer is formed from smaller units known as neurons. Neurons receive the input signals x and propagate them forward to the network and maps the complex relationship between inputs and the output. The first step in approximating the weight parameters of the model is finding the approximate architecture of the MLP, where the architecture is characterized by the number of hidden units, the type of activation function, as well as the number of input and output variables. The second step estimates the weight parameters using the training set [7]. Training estimates the weight vector $\vec{W}$ that ensures that the output is as close to the target vector as possible. This paper implements the autoencoder neural network as discussed below.

Autoencoder neural networks: Autoencoders, also known as auto-associative neural networks, are neural networks trained to recall the input space. Thompson *et al* [8] distinguish two primary features of an autoencoder network, namely the auto-associative nature of the network and the presence of a bottleneck that occurs in the hidden layers of the network, resulting into a butterfly-like structure. In cases where it is necessary to recall the input, autoencoders are preferred due to their remarkable ability to learn certain linear and non-linear interrelationships such as correlation and covariance inherent in the input space. Autoencoders project the input onto some smaller set by *intensively squashing* it into smaller details. The optimal number of the hidden nodes of the autoencoder, though dependent on the type of application, must be smaller than that of the input layer [8]. Autoencoders have been used in various applications including the treatment of missing data problem by a number of researchers including [4] and [9].

In this paper, auto-encoders are constructed using the MLP networks and trained using back-propagation. The structure of an autoencoder constructed using an MLP network is shown in Figure 1. The first step in approximating the weight parameters of the model is finding the approximate architecture of the MLP, where the architecture is characterized by the number of hidden units, the type of activation function, as well as the number of input and output variables. The second step estimates the weight parameters using the training set [7].

Training estimates the weight vector $\vec{W}$ to ensure that the output is as close to the target vector as possible. The problem of identifying the weights in the hidden layers is solved by maximizing the probability of the weight parameter using Bayes' rule [8] as follows:

$$p(\vec{W} \mid D) = \frac{P(D \mid \vec{W}) P(\vec{W})}{P(D)} \tag{1}$$

Where:

D is the training data, $P(\vec{W} \mid D)$ is the posterior probability, $P(D \mid \vec{W})$ is called the likelihood term that balances between fitting the data well and helps in avoiding overly complex models whereas $P(\vec{W})$ is the prior probability of $\vec{W}$ and $P(D)$ is the evidence term that normalizes the posterior probability. The input is transformed from x to the middle layer, a, using weights w_{ij} and biases b_i as follows [8]:

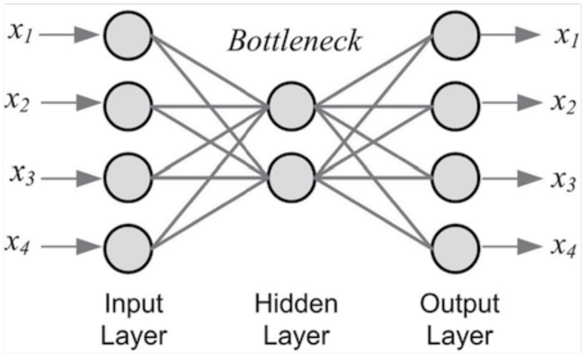


Figure 1: The structure of a four-input four-output autoencoder

$$a_j = \sum_{i=1}^{d} \vec{W}_{ji} x_i + b_j \qquad (2)$$

where j = 1 and j = 2 represent the first and second layer respectively. The input is further transformed using the activation function such as the hyperbolic tangent (*tanh*) or the sigmoid in the hidden layer. More information on neural networks can be found in [10].

3.2 Genetic Algorithms

Genetic algorithms use the concept of survival of the fittest over consecutive generations to solve optimisation problems [11]. As in biological evolution, the fitness of each population member in a generation is evaluated to determine whether it will be used in the breeding of the next generation. In creating the next generation, the use of techniques (such as inheritance, mutation, natural selection, and recombination) common in the field of evolutionary biology are employed. The GA algorithm implemented in this paper uses a population of string chromosomes, which represent a point in the search space [11]. In this paper, all GA parameters were empirically determined. GA is implemented by following three main procedures which are selection, crossover and mutation. The algorithm listing in Figure 2 illustrates how GA operates.

Fuzzy ARTMAP is a neural network architecture developed by Carpernter *et al* [12] and is based on Adaptive Resonance Theory (ART). The Fuzzy ARTMAP has been used in condition monitoring by Javadpour and Knapp [13], but their application was not online. The Fuzzy ARTMAP architecture is capable of fast, online, supervised incremental learning, classification and prediction [12]. The fuzzy ARTMAP operates by dividing the input space into a number of hyperboxes, which are mapped to an output space. Instance based learning is used, where each individual input is mapped to a class label. Three parameters namely the vigilance $\rho \in [0, 1]$, the learning rate $\beta \in [0, 1]$ and the choice parameter α, are used to control the learning process. The choice parameter is generally made small and a value of 0.01 was used in this application. The parameter β controls the adaptation speed, where 0 implies a slow speed and 1, the fastest. If $\beta = 1$, the hyperboxes get enlarged to include the point represented by the input vector. The vigilance represents the degree of belonging and it controls how large any hyperbox can become, resulting in new hyperboxes being formed. Larger values of ρ lead to a case where smaller hyperboxes are formed and this eventually leads to 'category proliferation', which can be viewed as overtraining. A complete description of the Fuzzy ARTMAP is provided in [12]. In this work, Fuzzy ARTMAP is preferred due to its incremental learning ability. As new data is sampled, there will be no need to retrain the network as would be the case with the MLP.

```
GA Algorithm
1). Create an initial population $P$, beginning at an initial
    generation $g = 0$.
2). for each population P, evaluate each population
    member (chromosome) using the defined fitness
    evaluation function possessing the knowledge of the
    competition environment.
3). using genetic operators such as inheritance,
    mutation and crossover, alter $P(g)$ to
    produce $P(g+1)$ from the fit chromosomes in P
    (g).
4). repeat steps (2) and (3) for the number of
    generations $G$ required.
```

Figure 2: Schematic representation of the Genetic algorithm operation

4. NEURAL NETWORKS AND GENETIC ALGORITHM FOR MISSING DATA

The method used here combines the use of auto-associative neural networks with genetic algorithms to approximate missing data. This method has been used by Abdella and Marwala [4] to approximate missing data in a database. A genetic algorithm is used in this work to *estimate* the missing values by optimising an objective function as presented shortly in this section. The complete vector combining the estimated and the observed values is input into the autoencoder as shown in Figure 3. Symbols X_k and X_u represent the known variables and the unknown (or missing) variables respectively. The combination of X_k and X_u represent the full input space.

Considering that the method proposed here uses an autoencoder, one will expect the input to be very similar to the output for a well chosen architecture of the autoencoder. This is, however, only expected on a data

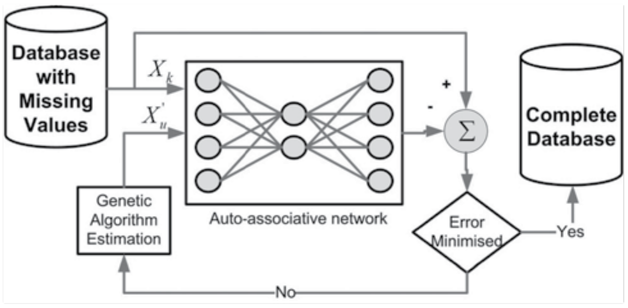


Figure 3: Autoencoder and GA Based missing data estimator structure

set similar to the problem space from which the inter-correlations have been captured. The difference between the target and the actual output is used as the error and this error is defined as follows:

$$\varepsilon = \vec{x} - f(\vec{W}, \vec{x}) \tag{3}$$

where $\vec{x}$ and $\vec{W}$ are input and weight vectors respectively. To make sure the error function is always positive, the square of the equation is used. This leads to the following equation:

$$\varepsilon = (\vec{x} - f(\vec{W}, \vec{x}))^2 \tag{4}$$

Since the input vector consist of both the known, X_k and unknown, X_u entries, the error function can be written as follows:

$$\varepsilon = \left(\begin{Bmatrix} X_k \\ X_u \end{Bmatrix} - f\left(\begin{Bmatrix} X_k \\ X_u \end{Bmatrix}, w \right) \right)^2 \tag{5}$$

and this equation is used as the objective function that is minimised using GA.

5. PROPOSED METHOD: ENSEMBLE BASED TECHNIQUE FOR MISSING DATA

The algorithm proposed here uses an ensemble of neural networks to perform both classification and regression in the presence of missing data. Ensemble based approaches have well been researched and have been found to improve classification performances in various applications [14-15]. The potential of using an ensemble based approach for solving the missing data problem remains unexplored in both classification and regression problems. In the proposed method, batch training is performed whereas testing is done online. Training is achieved using a number of neural networks, each trained with a different combination of features. For a condition monitoring system that contains n sensors, the user has to state the value of n_{avail}, which is the number of features most likely to be available at any given time. Such information can be deduced from the reliability of the sensors as specified by manufacturers. Sensor manufacturers often state specifications such as *Mean-time-between failures* (MTBF) and *Mean-time-to-failure* (MTTF) which can help in detecting which sensors are more likely to fail than others. MTTF is used in cases where a sensor is replaced after a failure, whereas MTBF denotes time between failures where the sensor is repaired. There is nevertheless, no guarantee that failures will follow manufacturers' specifications.

When the number of sensors most likely to be available has been determined, the number of all possible networks can be calculated using:

$$N = \begin{pmatrix} n \\ n_{avail} \end{pmatrix} = \frac{n!}{n(n - n_{avail})!} \tag{6}$$

where N is the total number of all possible networks, n is the total number of features and n_{avail} is the number of features most likely to be available at any time. Although the number n_{avail} can be statistically calculated, it has an effect on the number of networks that can be available. Let us consider a simple example where the input space has 5 feature, labelled: *a*, *b*, *c*, *d* and *e* and there are 3 features that are most likely to be available at any time. Using equation (6), variable N is found to be 10. These classifiers will be trained with features [*abc*, *abd*, *abe*, *acd*, *ace*, *ade*, *bcd*, *bce*, *bde*, *cde*]. In a case where one variable is missing, say, *a*, only four networks can be used for testing, and these are the classifiers that do not use *a* in their training input sequence. If we get a situation where two variables are missing, say *a* and *b*, we are left with one classifier. As a result, the number of classifiers reduces with an increase in the number of missing inputs per instance.

Each neural network is trained with n_{avail} features. The validation process is then conducted and the outcome is used to decide on the combination scheme. The training process requires complete data to be available as training is done off-line. The available data set is divided into the 'training set' and the 'validation set'. Each network created is tested on the validation set and is assigned a weight according to its performance on the validation set. A diagrammatic illustration of the proposed ensemble approach is presented in Figure 4.

For a classification task, the weight is assigned using the weighted majority scheme given by [16] as:

$$\alpha_i = \frac{1 - E_i}{\sum_{j=1}^{N} (1 - E_i)} \tag{7}$$

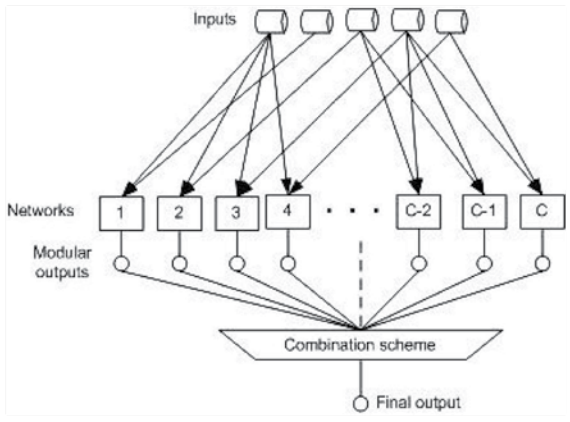


Figure 4: Diagrammatic illustration of the proposed ensemble based approach for missing data

where E_i is the estimate of model *i*'s error on the validation set. This kind of weight assignment has its roots in what is called boosting and is based on the fact that a set of networks that produces varying results can be combined to produce better results than each individual network in the ensemble [16]. The training algorithm is presented in *Algorithm 1* and the parameter *ntwk*$_i$ represents the *i*th neural network in the ensemble.

The testing procedure is different for classification and regression. In classification, testing begins by selecting an elite classifier. This is chosen to be the classifier with the best classification rate on the validation set. To this *elite* classifier, two more classifiers are gradually added, ensuring that an odd number is maintained. Weighted majority voting is used at each instance until the performance does not improve or until all classifiers are utilised. In a case of regression, all networks are used all at once and their predictions, together with their weights are used to compute the final value. The final predicted value is computed as follows:

$$f(x) = y \equiv \sum_{i=1}^{N} \alpha_i f_i(x) \qquad (8)$$

where α is the weight assigned during the validation stage when no data were missing and *N* is the total number of regressors. The parameter α is assigned such that $\sum_{i=1}^{N} \alpha_i = 1$. Considering that not all networks shall be available during testing, we define N_{usable} as the number of regressors that are usable in obtaining the regression value of an instance *j*. As a result $\sum_{i=1}^{N_{usable}} \alpha_i \neq 1$. We try to solve this by recalculating the weights such that the sum of all weights corresponding to N_{usable} is 1.

6. EXPERIMENTAL RESULTS AND DISCUSSION

This section presents the results obtained in the experiments conducted using the two techniques presented above. Firstly, the results of the proposed technique in a classification problem will be presented and later the method will be tested in a regression problem. In both cases, the results are compared to those obtained after imputing the missing values using the neural network-genetic algorithm combination as discussed above.

6.1 Application to classification

Data set: The experiment was performed using the Dissolved Gas Analysis (DGA) data obtained from a transformer bushing operating on-site. The data consist of 10 features, which are the gases that dissolved in the oil. The hypothesis in this experiment is to determine if the

```
Algorithm 1: Proposed algorithm for classification tasks
input  : all variable ∈ InputSpace & $n_{avail}$ obtained
         from the user
output: class
Calculate number of maximum Networks $N$ using Eq
(6)
forall ($variables \quad 1 \rightarrow X_n$) do
    Create all possible networks, $ntwk_1 \rightarrow ntwk_C$,
    each with $n_{avail}$ inputs
end
while Training do
    ← Train $ntwk_i$ with a different combination of $n_{avl}$
    inputs
    forall $i \rightarrow C$ do
        ← Subject $ntwk_i$ to a validation set as follows:
        ⟶ Select the corresponding features used;
        ⟶ Obtain network performance;
        ⟶ Assign weights, $\alpha$ according to Eq (7) and
        store for future use
    end
end
while Testing do
    ← Load parameters from trainning;
    if A Classification problem then
        foreach instance with missing values do
            ← Select networks, starting with those with
            bigger $\alpha$;
            ← Bring 2 more networks, using their $\alpha$as
            the selection criteria;
            ← Use majority voting to obtain the final
            classification
        end
    end
    if A Regression Problem then
        foreach instance with missing values do
            ← Get regression estimates from all
            networks trained without the current
            missing variable
            ← Use their weights to compute the final
            value.
        end
    end
end
```

Figure 5: Proposed algorithm for classification tasks

bushing condition (faulty or healthy) can be determined while some of the data are missing. The data was divided into the training set and the validation, each containing 2000 instances.

Experimental setup: The classification test was implemented using an ensemble of Fuzzy-ARTMAP networks. Two inputs were considered more likely to be missing and as a result, 8 were considered most likely to be available. The online process was simulated where data is sampled one instance at a time for testing. The network parameters were empirical determined and the vigilance parameter of 0.75 was used for the Fuzzy-ARTMAP. The results obtained were compared to those obtained using the the NN-GA approach, where for the GA, the crossover rate of 0.1 was used over 25 generations, each with a population size of 20. All these parameters were empirically determined.

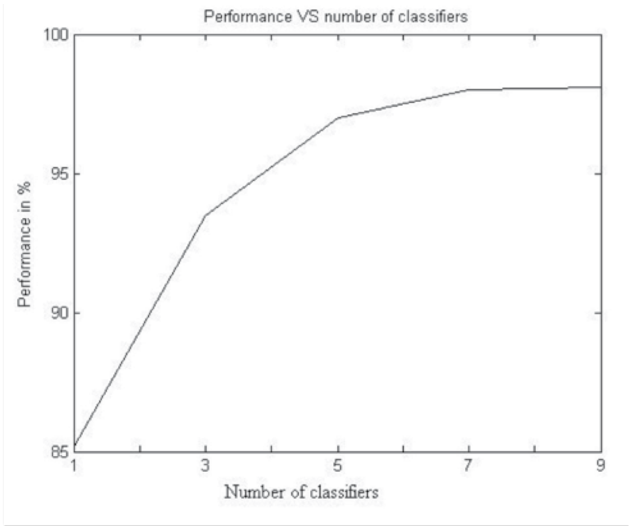


Figure 6: Diagrammatic illustration of the proposed ensemble based approach for missing data

Results: Using Equation (6), a total of 45 networks was found to be the maximum possible. The performance was calculated only after 4000 cases have been evaluated and is shown in Figure 6. The classification increases with an increase in the number of classifiers used. Although all these classifiers were not trained with all the inputs, their combination seems to work better than one network. The classification accuracy obtained under missing data goes as high as 98.2% which compares very closely to a 100 % which is obtainable when no data is missing.

Using the NN-GA approach, a classification of 96% was obtained. Results are tabulated in Table I below.

Table I: Comparison between the proposed method and the NN-GA approach

	Proposed Algorithm		NN-GA	
Number of missing	1	2	1	2
Accuracy (%)	98.2	97.2	99	89.1
Run time (s)	0.86	0.77	0.67	1.33

The results presented in Table I clearly show that the proposed algorithms can be used as a means of solving the missing data problem. The proposed algorithm compares very well to the well know NN-GA approach. The run time for testing the performance of the method varies considerably. It can be noted from the table that for the NN-GA method, run time increase with increasing number of missing variables per instance. Contrary to the NN-GA, our proposed method offers run times that decrease with increasing number of inputs. The reason for this is that the number of Fuzzy-ARTMAP networks available reduces with an increasing number of inputs as mentioned earlier. However, this improvement in speed comes at a cost of the diversity. We tend to have less diversity as the number of training inputs increase. Furthermore, this method will completely come to a failure in a case where more than n_{avl} inputs will be missing at the same time.

6.2 Application to regression

In this section, we extend the algorithm implemented in the above section to a regression problem. Instead of using an ensemble of Fuzzy ARTMAP networks as in classification, MLP networks are used. The reasons for this practice are two fold; firstly because MPL's are excellent regressors and secondly, to show that the proposed algorithm can be used with any architecture of neural networks.

Database: The data from a model of a Steam Generator at Abbott Power Plant [17] was used for this task. This data has four inputs, which are the *fuel, air, reference level* and the *disturbance* which is defined by the load level. There are two outputs which we shall try to predict using the proposed approach in the presence of missing data. These outputs are *drum pressure* and the *steam flow*.

Experimental setup: Although Fuzzy-ARTMAP could not be used for regression, we extended the same approach proposed above using MLP neural networks for regression problem. As before, this work regresses in order to obtain two outputs which are the *drum pressure* and the *steam flow*. We assume $n_{avl} = 2$ is the case and as a result, only two inputs can be used. We create an ensemble of MLP networks, each with five hidden nodes and trained only using two of the inputs to obtain the output. Due to limited features in the data set, this work shall only consider a maximum of one sensor failure per instance. Each network was trained with 1200 training cycles using the scaled conjugate gradient algorithm and a hyperbolic tangent activation function. All these training parameters were again empirically determined.

Table II: Regression accuracy obtained without estimating the missing values.

	Proposed Algorithm		NN-GA	
Number of missing	1	2	1	2
	Perf (%)	Time	Perf (%)	Time
Drum Pressure	98.2	97.2	68	126
Steam Flow	86	0.77	84	98

Results: Since testing is done online where one input arrives at a time, evaluation of performance at each instance would not give a general view of how the algorithm works. The work therefore evaluates the general performance using the following formula only after *N* instances have been predicted.

$$Error = \frac{n_\tau}{N} \times 100\% \tag{9}$$

where n_τ is the number of predictions within a certain tolerance. In this paper, a tolerance of 20% is used and was arbitrarily chosen. Results are summarised in Table II.

'Perf' indicates the accuracy in percentage whereas time indicates the running time in seconds. Results show that the proposed method is well suited for the problem under investigation. The proposed method performs better than the combination of GA and autoencoder neural networks in the regression problem under investigation. The reason is that the errors that are made when inputting the missing data in the NN-GA approach are further propagated to the output-prediction stage. The ensemble based approach proposed here does not suffer from this problem as there is no attempt to approximate the missing variables. It can also be observed that the ensemble based approach takes less time that the NN-GA method. The reason for this is that GA may take longer times to converge to reliable estimates of the missing values depending on the objective function to be optimised. Although, the prediction times are negligibly small, an ensemble based technique takes more time to train since training involves a lot of networks.

7. CONCLUSION

In this paper a new techniques for dealing with missing data for online condition monitoring problem was presented and studied. Firstly the problem of classifying in the presence of missing data was addressed, where no attempts are made to recover the missing values. The problem domain was then extended to regression. The proposed technique performs better than the NN-GA approach, both in accuracy and time efficiency during testing. The advantage of the proposed technique is that it eliminates the need for finding the best estimate of the data, and hence, saves time. Future work will explore the incremental learning ability of the Fuzzy ARTMAP in the proposed algorithm.

ACKNOWLEDGMENTS

The financial assistance of the National Research Foundation (NRF) of South Africa and the Carl and Emily Fuchs Foundation is hereby acknowledged.

8. REFERENCES

[1] R. J. A. Little and D. B. Rubin, *Statistical analysis with missing data*. New York: John Wiley, 1987.

[2] J. Kim and J. Curry, "The treatment of missing data in multivariate analysis," *Sociological Methods and Research*, vol. 6, pp. 215–241, 1977.

[3] J. Schafer and J. Graham, "Missing data: Our view of the state of the art," *Psychological Methods*, vol. 7, pp. 147– 177, 2002.

[4] M. Abdella and T. Marwala, "The use of genetic algorithms and neural networks to approximate missing data in database," *Computing and Informatics*, pp. 1001–1013, 2006.

[5] S. M. Dhlamini, F. V. Nelwamondo, and T. Marwala, "Condition monitoring of hv bushings in the presence of missing data using evolutionary computing," *WSEAS Transactions on Power Systems*, vol. 1, pp. 296–302, 2006.

[6] B. Gabrys, "Neuro-fuzzy approach to processing inputs with missing values in pattern recognition problems," *International Journal of Approximate Reasoning*, vol. 30, pp. 149–179, 2002.

[7] N. Japkowicz, "Supervised learning with unsupervised output separation," *In International Conference on Artificial Intelligence and Soft Computing*, pp. 321–325, 2002.

[8] B. B. Thompson, R. Marks, and M. A. El-Sharkawi, "On the contractive nature of autoencoders: Application to sensor restoration," *Proceedings of the IEEE International Joint Conference on Neural Networks*, pp. 3011– 3016, 2003.

[9] A. Frolov, A. Kartashov, A. Goltsev, and R. Folk, "Quality and efficiency of retrieval for willshaw-like autoassociative networks," *Computation in Neural Systems*, vol. 6, 1995.

[10] C. M. Bishop, *Neural Networks for Pattern Recognition*. New York: Oxford University Press, 2003.

[11] D. Goldberg, *Genetic Algorithms in Search, Optimization and Machine Learning*. Reading, MA: Addison-Wesley, 1989.

[12] G. Carpenter, S. Grossberg, N. Markuzon, J. Reynolds, and D. Rosen, "Fuzzy ARTMAP: A neural network architecture for incremental supervised learning of multidimensional maps," *IEEE Transactions on Neural Networks*, vol. 3, pp. 698–713, 1992.

[13] R. Javadpour and G. Knapp, "A fuzzy neural network approach to condition monitoring," *Sociological Methods and Research*, vol. 45, pp. 323–330, 2003.

[14] Y. Freund and R. Schapire, "A decision theoretic generalization of online learning and an application to boosting," in *Proceedings of the Second European Conference on Computational Learning Theory*, pp. 23–37, 1995.

[15] L. I. Kuncheva, *Combining Pattern Classifies, Methods and Algorithms*. New York: Willey Interscience, 2005.

[16] E. McGookin and D. Murray-Smith, "Using correspondence analysis to combine classifiers," *Machine Learning*, vol. 14, pp. 1–26, 1997.

[17] B. De Moor, "Database for the identification of systems, department of electrical engineering, esat/sista." Internet Listing, Last Acessed: 2 April 2006 1998.URL: http://www.esat.kuleuven.ac.be/sista/daisy.

ON VISUAL OBJECT TRACKING USING ACTIVE APPEARANCE MODELS

M.R. Hoffmann, B.M. Herbst, and K.M. Hunter

Applied Mathematics, University of Stellenbosch, Private Bag X1, Matieland, 7602, South Africa

Abstract: Active appearance models provide an elegant framework for tracking objects. Using them in a deterministic algorithm to perform tracking is not robust enough since no history is used of the object's movement and position. We discuss two approaches to rectify this situation. Both techniques are based on the particle filter. The first technique initialises the active appearance model search algorithm with a shape estimate obtained from an active contour tracker. A combination of a particle filter and an active appearance model forms the foundation for the second technique. Experimental results indicate the effectiveness of these techniques.

Key Words: active appearance models, particle filter, tracking

1. INTRODUCTION

Active appearance models (AAMs) [1] provide a neat model-based framework for tracking objects. They incorporate both shape and texture into their formulation, hence they enable us to track simultaneously the outline of an object as well as its appearance. It is therefore easy to use the parameters provided by an AAM tracker in other applications.

Stegmann [2] demonstrated that AAMs can be successfully applied to perform object tracking. In his deterministic approach, the AAM search algorithm is applied successively to each frame. However, this technique is not robust since the optimisation techniques employed in the AAM search algorithm only explores a small, local region of interest. No history of the object's movement and position is used to improve the optimisation. Therefore the tracker fails when the object moves fast for example. The reason for this failure is that the sudden jumps caused by fast movements lead to a bad initialisation for the AAM optimisation routines.

In this paper, two techniques are discussed which can be used to make the deterministic AAM tracker more robust. The first technique is based on active contours. It initialises the AAM using the shape estimate obtained from active contours. In this way, the AAM search algorithm is narrowed down to a local region of interests around the estimate from active contours. Since the active contours algorithm is able to track an object robustly, it is a better initialisation for the AAM. The second technique uses a combination of a particle filter and an AAM to provide more robustness. Here temporal filtering predicts the parameters of the AAM so that the history of the object's movement and position enhances the AAM searches.

Central to both techniques is a particle filter. Particle filters have become an important tool to track objects. They have been used in conjunction with edge measurements (active contours) [3], colour histograms [4, 5] and even a combination of the aforementioned [6]. Particle filters [7, 8] can deal with non-linear systems and non-Gaussian models, and can therefore be viewed as a generalisation of the Kalman filter.

The rest of the paper is organised as follows: we briefly summarise active appearance models in Section 2, followed by a summary of particle filters in Section 3. The first technique to enhance the standard AAM tracker is detailed in Section 4, and the second technique in Section 5. Experimental results are shown in Section 6. We conclude in Section 7.

2. ACTIVE APPEARANCE MODELS (AAMS)

Active appearance models [1, 9] are template based and employ both shape and texture. Using them consists of an initial training phase to learn the parameters for an object (e.g. $\boldsymbol{\Phi}_s$, $\overline{\boldsymbol{s}}$ in Equation 1 below) and a search phase to extract the object in a new image.

In the training phase of AAM, the shape of the modelled object is defined by a vector of feature points on the outline of the object, $\boldsymbol{s}_i = [x_{i1}, \cdots, x_{in}, y_{i1}, \cdots, y_{in}]^T$, $i = 1, \ldots, l$ where l is the number of training images and n the number of feature points. The shapes are normalised with respect to translation, scale and rotation and this is done by setting up a common coordinate reference. Then all the shapes are aligned to this reference giving rise to a parameter known as the pose $\boldsymbol{p}$. Using the pose and the common coordinate reference, an aligned shape (now in relative coordinates) can be translated, scaled and rotated to its original version (in absolute coordinates). The texture of the object in question is described by a vector, $\boldsymbol{g}_i = [g_{i1}, g_{i2}, \ldots, g_{im}]^T$, $i = 1, \ldots, l$ with m the number of texture points.

Typically a piece-wise affine warp based on the Delaunay triangulation is used to collect the texture points. Principal component analysis (PCA) is performed on the aligned shapes and textures and this yields

$$\boldsymbol{s} = \overline{\boldsymbol{s}} + \boldsymbol{\Phi}_s \boldsymbol{b}_s \quad (1)$$
$$\boldsymbol{g} = \overline{\boldsymbol{g}} + \boldsymbol{\Phi}_g \boldsymbol{b}_g \quad (2)$$

where $\boldsymbol{\Phi}_s$ and $\boldsymbol{\Phi}_g$ are the eigenvector matrices of the shape and texture covariance matrices respectively and $\boldsymbol{b}_s$ and $\boldsymbol{b}_g$ are the PCA projection coefficients.

A combined model parameter $\boldsymbol{c}$ is obtained by combining the PCA scores into $\boldsymbol{b} = [\boldsymbol{\Psi}_s \boldsymbol{b}_s,\ \boldsymbol{b}_g]^T$ and performing a third PCA

$$\boldsymbol{b} = \boldsymbol{\Phi}_c \boldsymbol{c}, \quad (3)$$

where $\boldsymbol{\Psi}_s$ is a weighting matrix between pixel intensities and pixel distances and $\boldsymbol{\Phi}_c$ is the basis for the combined model parameter space. Writing $\boldsymbol{\Phi}_c = \left[\boldsymbol{\Phi}_{c,s}, \boldsymbol{\Phi}_{c,g}\right]^T$, it is now possible to generate new shapes and texture instances by

$$\boldsymbol{s} = \overline{\boldsymbol{s}} + \boldsymbol{\Phi}_s \boldsymbol{\Psi}_s^{-1} \boldsymbol{\Phi}_{c,s} \boldsymbol{c} \quad (4)$$
$$\boldsymbol{g} = \overline{\boldsymbol{g}} + \boldsymbol{\Phi}_g \boldsymbol{\Phi}_{c,g} \boldsymbol{c}. \quad (5)$$

In the search phase of AAMs, the model parameter $\boldsymbol{c}$ and the pose $\boldsymbol{p}$ are sought that best represent an object in a new image not contained in the original training set. In practise changing $\boldsymbol{c}$ varies both the texture and the shape of an object. Suppose we need to find an object in an image. The idea is to vary $\boldsymbol{c}$ (optimise over $\boldsymbol{c}$) so that the shape and texture generated by Equation 4 and Equation 5 fits the object in the image as well as possible. The objective function that is minimised is the difference

$$E = ||\boldsymbol{g}_{model} - \boldsymbol{g}_{image}||^2 = ||\delta \boldsymbol{g}||^2 \quad (6)$$

between the texture values generated by $\boldsymbol{c}$ and Equation 5, denoted as $\boldsymbol{g}_{model}$, and the texture values in the image, $\boldsymbol{g}_{image}$. Note that the image texture values $\boldsymbol{g}_{image}$ for a specific value of $\boldsymbol{c}$ are the values sampled from the shape generated by Equation 4 and then translated, scaled and rotated using the pose $\boldsymbol{p}$. This Euclidean transformation is necessary since the shape generated by Equation 4 is in the normalised coordinate system and we need to sample the texture values in the image coordinate system. In summary, the optimisation over $\boldsymbol{c}$ minimises Equation 6, i.e. produces the best fit of texture values.

For the implementation AAMs assumes that there exists a linear relationship between the differences in texture values $\delta \boldsymbol{g} = \boldsymbol{g}_{model} - \boldsymbol{g}_{image}$ and the model parameters' update, hence

$$\delta \boldsymbol{p} = \boldsymbol{R}_p \delta \boldsymbol{g} \quad (7)$$
$$\delta \boldsymbol{c} = \boldsymbol{R}_c \delta \boldsymbol{g}, \quad (8)$$

where $\boldsymbol{R}_p$ and $\boldsymbol{R}_c$ are found by conducting a set of experiments and using the results to perform multivariate linear regression. The parameters are fine-tuned by gradient-descend optimisation strategies.

The optimisation strategy described above, requires a good initialisation for the following reasons:

- The shape generated by $\boldsymbol{c}$ and Equation 4 is translated, scaled and rotated using $\boldsymbol{p}$—a large space to search.
- The assumption that there exists a linear relationship between the differences in texture values and the model parameters' updates is only reasonable for small updates.

From the discussion, we see that AAMs provide a general framework to track or segment different types of objects. Furthermore, no parameters need to be specified by an expert to use them. On the downside, AAM requires objects to have distinct features/outlines and there is a training phase involved. Also, a good initialisation is required for the search algorithm.

We used the open-source AAM-API [10] in our implementation. For further detail on AAMs, the reader may refer to [1, 9].

3. THE PARTICLE FILTER

Particle filters (also known as bootstrap filtering and the condensation algorithm) is a Monte Carlo-type technique to approximate probabilistic density functions (pdfs).

In the spirit of [3, 8], the state vector $\boldsymbol{x}_t \in \mathbb{R}^{n_x}$ describes the object to be tracked at time step t, while the measurements are given by $\boldsymbol{z}_t \in \mathbb{R}^{n_z}$. We denote all the measurements up until the tth time step by $\boldsymbol{Z}_t \triangleq \{\boldsymbol{z}_i, i = 1, \ldots, t\}$. In the Bayesian framework, the goal is to find an estimate of $\boldsymbol{x}_t$ based on all the observations $\boldsymbol{Z}_t$. Thus the conceptual Bayes solution recursively updates the posterior pdf

$$p(\boldsymbol{x}_t|\boldsymbol{Z}_t) = \frac{p(\boldsymbol{z}_t|\boldsymbol{x}_t, \boldsymbol{Z}_{t-1})\, p(\boldsymbol{x}_t|\boldsymbol{Z}_{t-1})}{p(\boldsymbol{z}_t|\boldsymbol{Z}_{t-1})} \quad (9)$$

as the measurements become available. In the particle filter, the posterior pdf is approximated by a weighted random sample set $\{\boldsymbol{x}_t^i, \pi_t^i\}_{i=1}^N$, $\sum_i \pi_t^i = 1$ where $\{\boldsymbol{x}_t^i, i = 1 \ldots N\}$ is a random measure of support points associated with the weights $\{\pi_t^i, i = 1 \ldots N\}$.

The particle filter algorithm consists of three phases that are repeated at each time step and are illustrated in Figure 1. The *selection* phase chooses particles according to their relative probabilities, i.e. particles with larger weights will be chosen several times, and those with smaller weights will be selected fewer times or discarded. During the *prediction* phase the selected particles are put through the process model to generate a prior pdf $p(\boldsymbol{x}_t|\boldsymbol{x}_{t-1})$. To account for process

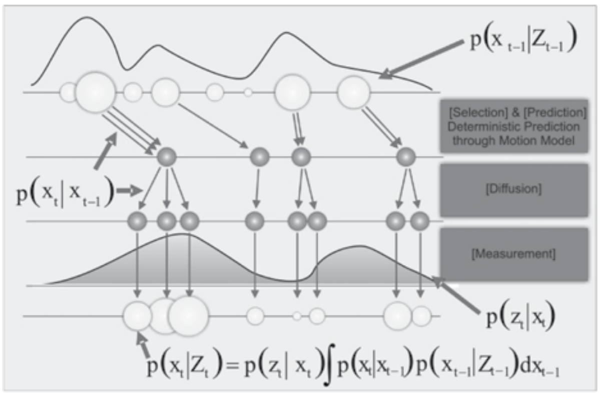

$p(x_{t-1}|Z_{t-1})$
[Selection] & [Prediction] Deterministic Prediction through Motion Model
$p(x_t|x_{t-1})$
[Diffusion]
[Measurement]
$p(z_t|x_t)$
$p(x_t|Z_t) = p(z_t|x_t)\int p(x_t|x_{t-1})p(x_{t-1}|Z_{t-1})dx_{t-1}$

Figure 1: A graphical illustration of one iteration of the particle filter. (Image courtesy of S.Fleck [11].)

noise, the particles are *diffused* by adding noise to them. The *measurement* phase updates the weights in the light of the new measurement $\boldsymbol{z}_t$

$$\pi_t^i = \frac{p(\boldsymbol{z}_t|\boldsymbol{x}_t = \boldsymbol{x}_t^i)}{\sum_{n=1}^{N} p(\boldsymbol{z}_t|\boldsymbol{x}_t = x_t^n)}. \tag{10}$$

4. ACTIVE CONTOURS AND ACTIVE APPEARANCE MODELS

The combined active contour and active appearance model (AC-AAM) finds a shape to initialise the AAM algorithm using active contours and was proposed by Sung & Kim [12]. This method rectifies the situation in which AAM only works locally well. A summary of this technique is presented below.

4.1. *Contours and shape space*

B-splines, see [7, 13] for example, are used to model the outline of the tracked object. Given a set of coordinates of control points $(x_1, y_1), \ldots, (x_n, y_n)$, a B-spline is the curve $(\boldsymbol{r}(s) = (x(s), y(s))$ formed by a parametrisation with parameter s on the real line,

$$\boldsymbol{r}(s) = \begin{bmatrix} \boldsymbol{B}(s)^T & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{B}(s)^T \end{bmatrix} \begin{bmatrix} \boldsymbol{Q}^x \\ \boldsymbol{Q}^y \end{bmatrix} \tag{11}$$

where $\boldsymbol{B}(s)$ is the $n \times 1$ vector of B-spline basis functions, and $\boldsymbol{Q}^x$, $\boldsymbol{Q}^y$ are the vectors of control points determined by listing all the x-coordinates and y-coordinates respectively. We refer to a curve as defined by Equation 11 as a *contour*.

The dimension of the vector space spanned by $\boldsymbol{r}(s)$ is $N_Q = 2N_B$, where N_B is the number of control points of the B-spline. This implies N_Q degrees of freedom and it means that the object can deform in N_Q different ways if we track it over different frames. This amount of allowable deformation leads to many tracking errors. To rectify this situation, the allowable deformation is restricted to a lower dimensional space, known as the shape space.

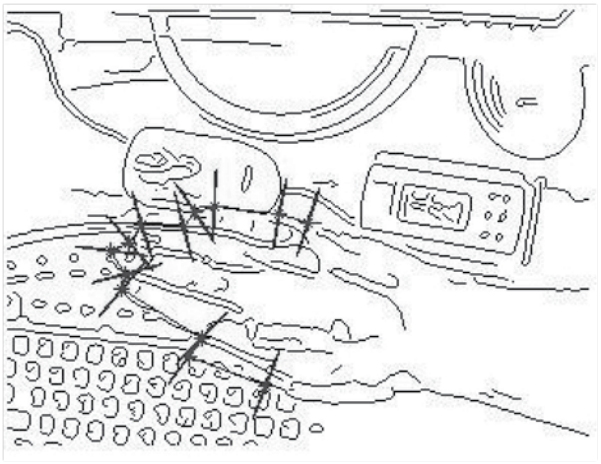


Figure 2: An example output from the Canny edge detector with a contour and its normal lines.

The shape space is defined as a linear mapping from a shape vector $\boldsymbol{X} \in R^{N_x}$ to a spline vector $\boldsymbol{Q} = \left[\boldsymbol{Q}^x,\ \boldsymbol{Q}^y\right]^T \in \mathbb{R}^{N_Q}$ and the mapping is given by

$$\boldsymbol{Q} = W\boldsymbol{X} + \boldsymbol{Q}_0 \tag{12}$$

where W is a matrix with rank $N_X \ll N_Q$, describing the allowable transformations. Variations are compared against the template curve $\boldsymbol{Q}_0$. By restricting $\boldsymbol{X}$, we can clearly restrict the transformations away from $\boldsymbol{Q}_0$.

4.2. *Active contours*

Following Blake & Isard [3], we summarise the contour based particle filter. Adapting a particle filter for a particular implementation, requires the specification of the state vector, process model and measurement model.

The state vector: The state vector at each time step t is given by the shape space vector. Thus $\boldsymbol{x}_t = \boldsymbol{X}_t$. This allow us to generate a vector of B-spline control points $\boldsymbol{Q}$ for each particle using Equation 12.

The process model: States evolve according to a simple random walk given by

$$\boldsymbol{x}_t^i = \boldsymbol{x}_{t-1}^i + \boldsymbol{S}_t^i\boldsymbol{u}_t^i \tag{13}$$

where $\boldsymbol{S}_t^i$ is the process noise covariance and $\boldsymbol{u}_t^i$ is a vector of normal distributed random variables. One can use more sophisticated process models and the reader is referred to [7] for a detailed discussion.

The measurement model: The binary edge map for the current frame is given to the algorithm to estimate the weight associated with each particle. An example of such an edge map, with a contour, its control points and normal lines superimposed on the edge map, are illustrated in Figure 2. To calculate the weight for the ith particle, the following procedure is followed:

- Using Equation 12 and the state vector, calculate the vector of control points $\boldsymbol{Q}^i$.
- Calculate the normal lines for each control point.
- For each control point, search along the normal line until an edge is found. Let the distance from the control points to the edge be $d_j, j = 1, \ldots, n$ where n is the number of control points. If an edge is not found, set the distance equal to the length of the normal line. Calculate the total length $d^i = \sum_{j=1}^{n} d_j$.
- The weight for the ith particle is then given by

$$\pi_i = \exp\left(-\frac{(d^i)^2}{\sigma^2}\right). \tag{14}$$

The weights are normalised afterwards to sum to unity. From equation Equation 14, we see that a small value of d^i will result in a larger value of π_i and vice versa. Furthermore, the variance σ determines how much preference we give to particles with a lower distance, d^i, or not.

4.3. *Initialisation of AAM with AC*

The AC-AAM tracker consists of two parts. In particular, at time step t, the first part performs standard active contour tracking and let the estimate from the tracker be $\boldsymbol{x}_t^e$. In the second part, Equation 12 is used together with $\boldsymbol{x}_t^e$ to generate a shape estimate

$$\boldsymbol{Q}_t^e = W\boldsymbol{x}_t^e + \boldsymbol{Q}_0. \tag{15}$$

Notice that $\boldsymbol{Q}_t^e$ and the AAM shape representation, $\boldsymbol{s}$, (not normalised with respect to the pose) are equivalent. This shape, $\boldsymbol{Q}_t^e$, is then used to initialise standard AAM optimisation and the result is output as the best fitted AAM. Sung & Kim [12] use the result of the AAM to initialise the active contour tracker again, but we find the aforementioned scheme adequate.

This technique uses the particle filter indirectly. The particle filter is an integral part of the active contour tracker, but the AAM part of the AC-AAM tracker does not utilise the particle filter.

5. AN ACTIVE APPEARANCE MODEL BASED PARTICLE FILTER

This section elucidates the second approach in order to increase the robustness of the AAM tracker. It is based on a direct combination of an AAM with a particle filter and was introduced by Hamlaoui & Davoine [14] with some additions made by Fleck et al [15]. It differs from the previous technique in the sense that the AAM is not initialised by a secondary technique, but instead temporal filtering predicts the parameters of the AAM. This way, history of the object improves the overall robustness of the AAM.

As in the case of the active contour tracker, the adaption of the particle filter to work in conjunction with an AAM, requires the specification of the state vector, the process model and the measurement model.

5.1. *The state vector*

The state vector is a combination of the model parameters $\boldsymbol{c}$ and the pose $\boldsymbol{p}$ and at time step t it is given by

$$\boldsymbol{x}_t = \begin{bmatrix} \boldsymbol{p} \\ \boldsymbol{c} \end{bmatrix}. \tag{16}$$

From this it is clear, that one can synthesise a shape and texture for a particular image given the model parameters.

5.2. *The process model*

The states evolves according to

$$\boldsymbol{x}_t^i = \hat{\boldsymbol{x}}_{t-1} + \begin{bmatrix} \boldsymbol{R}_p \\ \boldsymbol{R}_c \end{bmatrix} \delta\boldsymbol{g}_{t-1} + \boldsymbol{S}_t^{(i)}\boldsymbol{u}^{(i)} \tag{17}$$

where $\hat{\boldsymbol{x}}_{t-1}$ is the estimate of the state vector at time step $t-1$, $\boldsymbol{S}_t^{(i)}$ is the process noise covariance and $\boldsymbol{u}^{(i)}$ is a vector of normally distributed random variables.

5.3. *The measurement model*

Since the purpose of the measurement model is to classify how good a particular particle fits the underlying image, the optimisation criterion (equation Equation 6) will be used. Hence, for each particle the aforementioned norm is calculated and the pre-normalised weight is then given by

$$\pi_t^i = \exp\left(-\frac{E_i^2}{\sigma^2}\right) \tag{18}$$

where σ plays the same role as in the case with the active contour particle filter.

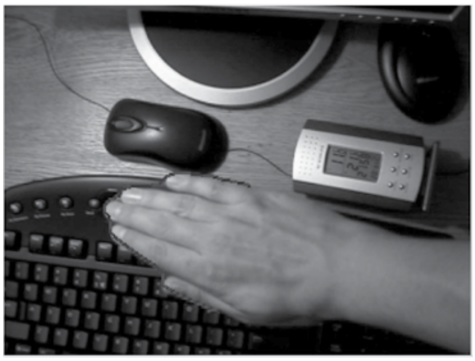
6. EXPERIMENTAL RESULTS

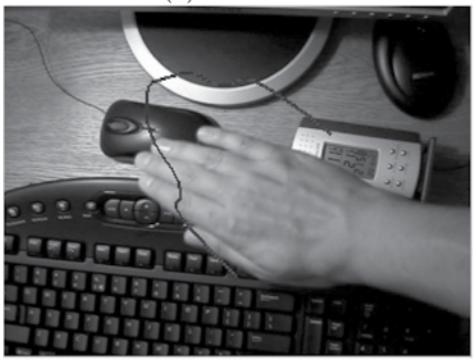
We implemented the AAM-based particle filter tracker in C++. The active contour implementation was done in MATLAB. To illustrate the effectiveness of the trackers, they were used to track a hand moving against a cluttered background and the results are available for download from the project's website [16]. The model of the hand consists of 13 feature points and the AAM was trained using 4 images.

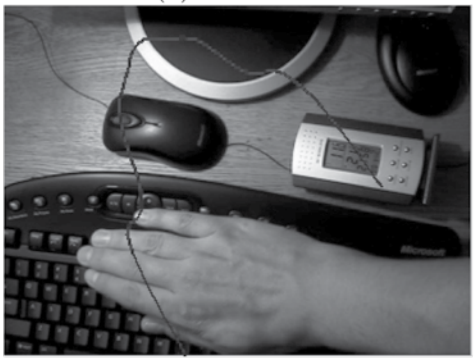
Tracking results of performing deterministic AAM tracking is shown in Figure 3. It is clear that this approach cannot handle fast movements well as explained in Section 4.

In Figure 4 the results obtained with the AC-AAM tracker are shown, while the corresponding results obtained with the AAM-based particle filter are illustrated in Figure 5. Both trackers are able to track the complete movement of the hand accurately.

When the AC-AAM approach is used, a simple dynamic model for the active contour tracker suffices. This can be seen in Figure 4 where the output of the


(a) Frame 10


(b) Frame 149


(c) Frame 184

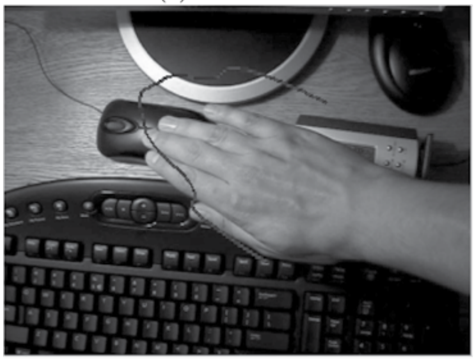
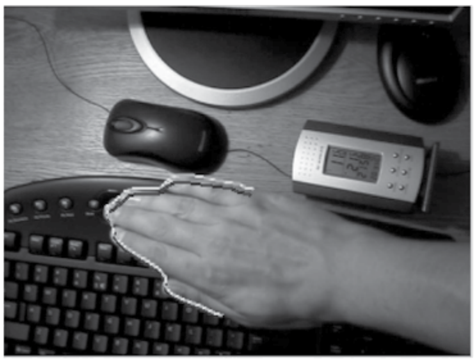
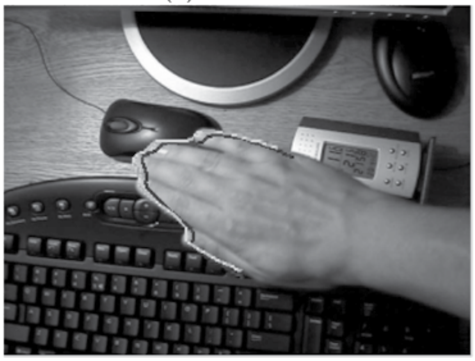
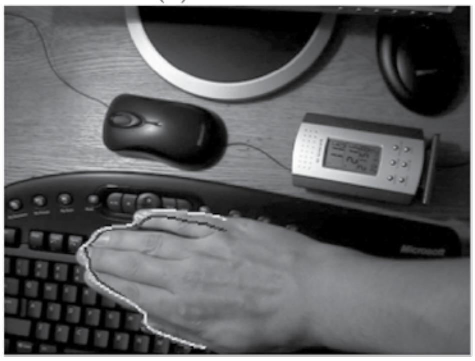

(d) Frame 310

Figure 3: A random selection of frames to indicate the performance of the deterministic AAM tracker. In (b), (c) and (d) the tracker loses its target due to fast movements.


(a) Frame 10


(b) Frame 149


(c) Frame 184

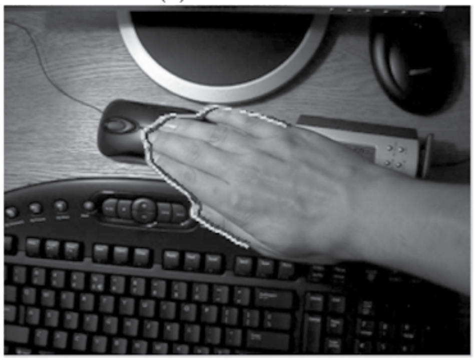
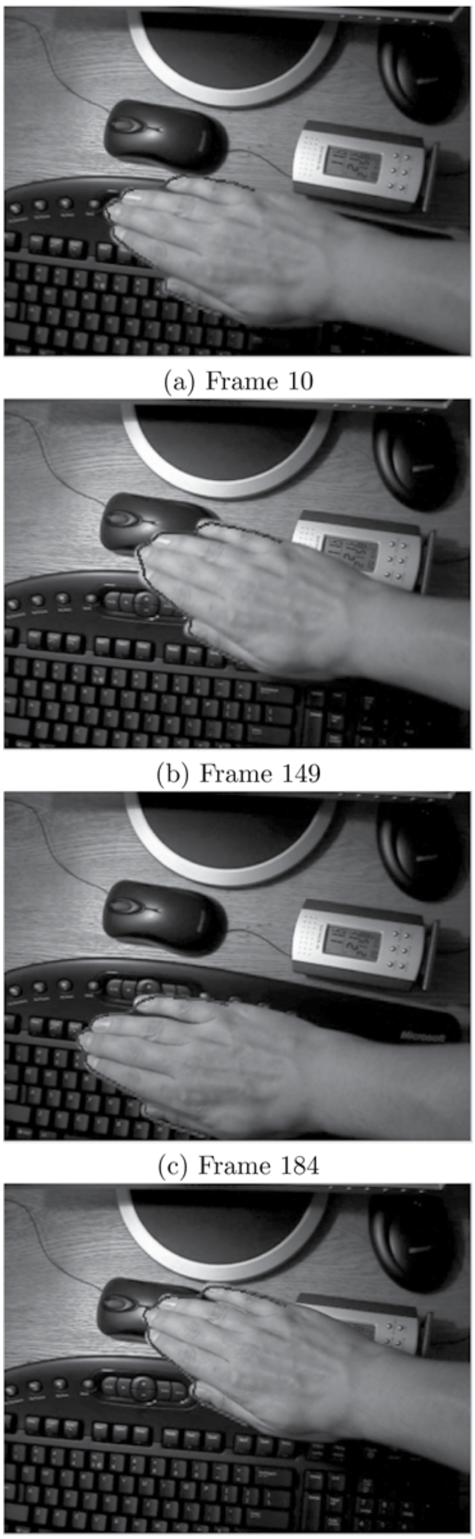

(d) Frame 310

Figure 4: A selection of frames to indicate the results of the AC-AAM tracker. The blue shape is the active contour's estimate and the green is the result after applying the AAM search algorithm, initialised with the blue shape.


(a) Frame 10

(b) Frame 149

(c) Frame 184

(d) Frame 310

Figure 5: Selected frames to indicate the performance of the AAM-based particle filter tracker.

active contour tracker is not as accurate, but the resulting AAM fit is. This illustrates the principal idea of the AC-AAM approach: the robustness of the active contour tracker is used to initialise the AAM which in turn, adjusts well to the underlying image.

7. CONCLUSION

We discussed two techniques to enhance the deterministic AAM tracker. Central to both techniques is the particle filter. The effectiveness of the trackers was illustrated by presenting the tracking results of an object moving against a cluttered background.

Using the AAM-based particle filter has the advantage that extreme deformations can be learned in a single step. If this is require for the AC-AAM tracker (we can only handle transformations up to an affinity), the active contour as well as the AAM must be trained.

We expect that the runtime of the AC-AAM will be better compared to the AAM-based particle filter and we are currently working on creating a common framework to verify this claim. Future work includes the investigation of techniques to deal with occlusions in the AC-AAM tracker.

Acknowledgements: The financial assistance of the South African National Research Foundation (all authors) and Schlumberger (M. Hoffmann) towards this research is gratefully acknowledged. Opinions expressed and conclusions arrived at, are those of the authors and are not necessarily to be attributed to the NRF and Schlumberger.

8. REFERENCES

[1] T. F. Cootes, G. J. Edwards, and C. J. Taylor, "Active Appearance Models," in *European Conference on Computer Vision*, vol. 2, pp. 484–498, 1998.

[2] M. B. Stegmann, "Object tracking using Active Appearance Models," in *Proc. 10th Danish Conference on Pattern Recognition and Image Analysis* (S. I. Olsen, ed.), vol. 1, (Copenhagen, Denmark), pp. 54–60, DIKU, Jul 2001.

[3] M. Isard and A. Blake, "Condensation – conditional density propagation for visual tracking," *International Journal of Computer Vision*, vol. 29, no. 1, pp. 5–28, 1998.

[4] P. Pérez, C. Hue, J. Vermaak, and M. Gangnet, "Color-based probabilistic tracking," in *ECCV '02: Proceedings of the 7th European Conference on Computer Vision-Part I*, (London, UK), pp. 661–675, Springer-Verlag, 2002.

[5] K. Nummiaro, E. Koller-Meier, and L. Van Gool, "A color-based particle filter," in *First International Workshop on Generative-Model-Based Vision GMBV'02, in conjuction with ECCV'02*, pp. 53–60, 2002.

[6] M. Spengler and B. Schiele, "Towards robust multi-cue integration for visual tracking," in *ICVS '01: Proceedings of the Second International Workshop on Computer Vision Systems*, (London, UK), pp. 93–106, Springer-Verlag, 2001.

[7] A. Blake and M. Isard, *Active Contours*. Springer-Verlag, London, 1998.

[8] B. Ristic, S. Arulampalam, and N. Gordon, *Beyond the Kalman Filter—Particle Filters for Tracking Applications*. Artech House, 1st ed., 2004.

[9] M. B. Stegmann, "Active Appearance Models: Theory, extensions and cases," Master's thesis, Informatics and Mathematical Modelling, Technical University of Denmark, DTU, Richard Petersens Plads, Building 321, DK-2800 Kgs. Lyngby, Aug 2000.

[10] M. B. Stegmann, B. K. Ersbøll, and R. Larsen, "FAME - a flexible appearance modelling environment," *IEEE Transactions on Medical Imaging*, vol. 22, no. 10, pp. 1319–1331, 2003.

[11] S. Fleck, S. Lanwer, and W. Straßer, "A smart camera approach to real-time tracking," in *13th European Signal Processing Conference (EUSIPCO 2005)*, September 4-8 2005.

[12] J. Sung and D. Kim, "A background robust Active Appearance Model using active contour technique," *The journal of the Pattern Recognition Society*, 2006. In press.

[13] B. Herbst and B. Fornberg, "Modelling in Applied Mathematics." [Online]. Available: `http://dip.sun.ac.za/courses/TW424/`, 2003.

[14] S. Hamlaoui and F. Davoine, "Facial action tracking using particle filters and Active Appearance Models," in *The Smart Objects and Ambient Intelligence conference*, October 2005.

[15] S. Fleck, M. Hoffmann, K. Hunter, and A. Schilling, "PFAAM - An Active Appearance Model based Particle Filter for both Robust and Precise Tracking," *The Fourth Canadian Conference on Computer and Robot Vision (CRV 2007) Montreal, Canada,*, May 2007.

[16] M. Hoffmann, "Project's website." `http://dip.sun.ac.za/~mcelory/vision-research/aam-tracking/`, 2006.

IDENTITY CONFIDENCE ESTIMATION OF MANOEUVRING AIRCRAFT

P.J. Holtzhausen, B.M. Herbst

Applied Mathematics, University of Stellenbosch, Private Bag X1, 7602, Matieland, South Africa

Abstract: A radar system observes an aircraft once during each scan of the airspace, and uses these observations to construct a track representing a possible route of the aircraft. However when aircraft interact closely there is the possibility of confusing the identities of the tracks. In this study multiple hypothesis techniques are applied to extract an identity confidence from a track, given a set of possible tracks and observations. The system utilises numerous estimation filters internally and these are investigated and compared in detail. The Identity Confidence algorithm is tested using a developed radar simulation system, and evaluated sucessfully against a series of benchmark tests.

Key Words: radar, tracking, identity, confidence

1. INTRODUCTION

Radar operates in a noisy world. It is the task of radar tracking software to keep track of an airplane, given noisy measurements and aircraft estimates. This is difficult since the measurements from different airplanes can be mixed and *false detections* are also a possibility. Sometimes the target remains undetected for undetermined lengths of time causing *missed detections*.

It is therefore unrealistic to expect a tracker to operate without error indefinitely, and identity checking mechanisms are used to ascertain that the aircraft that enters the airspace is in fact the aircraft that lands. In aerospace, the airplane identity is usually confirmed with the use of transponders or by means of radio communication. In military situations however, such aircraft identification is not always possible.

It is sometimes unavoidable for airplanes to manoeuvre close to one another, causing situations where identity confirmation is not straightforward. In combat situations radio silence is often enforced, and visual inspection often involves close quarters flight patterns.

Therefore in case of non-operational transponders, it is useful to determine to what extent two closely encountering aircraft might be confused with one another. Figure 1 illustrates a scenario involving two confirmed airplanes A and B with end positions supplied by a tracker. The tracker decided in this case that plane A moved to position A and plane B to position B, but the situation might have been indeed the opposite.

For a radar operator looking at aircraft interacting on the screen it can be useful to have an analysis tool that shows the identity confidence probability for a specific track. This can offer useful supplemental advice to aid human judgement during difficult decision making.

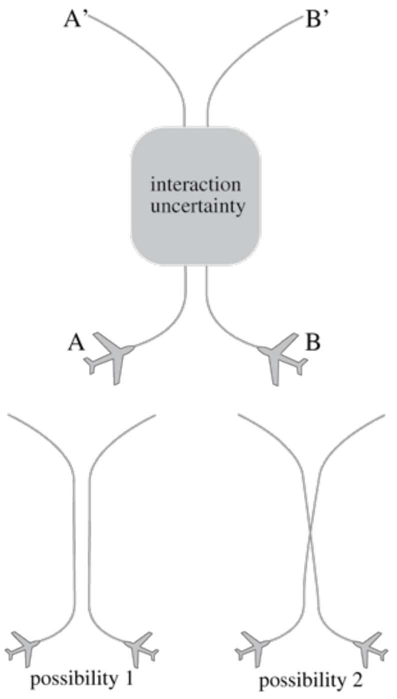


Figure 1: The interaction of two flight paths

The following factors play a role in this problem:

- **History**: The movement history of an airplane is important, and can be used to predict future positions. Past behaviour is also a good indication of future behaviour, for example an airplane starting a manoeuvre is more likely to be unpredictable than one flying a straight path.
- **Dynamics**: When engaged in an manoeuvre, aircraft dynamics can restrict the kind of motion that is achiev-

able. A pilot can make a 5g turn in combat, but anything higher is likely to be to his detriment.

- **Radar characteristics**: The radar site location and noise parameters largely determine the extent of the confusion. A radar typical makes greater azimuthal error than range error.

Approaching the problem from the radar tracking side, many of these aforementioned factors can be easily integrated. Kalman filters for example are designed to cope with noise corrupted measurements.

Radar tracking is a field with many methodologies. The simplest is the Global Nearest Neighbour (GNN) technique that *gates* (i.e. selects) measurements around each track, and then associates each measurement with a track to minimize the sum of measurement-to-track distances.

Bar Shalom [1] is a proponent of Probabilistic Data Association (PDA), where every track is updated by a weighted sum of all observations within the gating distance. Special attention needs to be paid to creation of new tracks and track interaction.

Reid [2] introduced the Multiple Hypothesis Tracker (MHT) that operates by considering every possibility of data association, and assigning a probability to each hypothesis. Instead of making a hard decision like the other techniques, the possibilities are propagated into the future with the idea that future data will resolve uncertainties. In a MHT hypothesis an estimation filter is assigned to each aircraft to give the best possible estimate of position and velocity.

After first covering general radar background, we will investigate estimation filters in the **Track Modeling** section after which it will be integrated with the MHT in the **Track Management** section. Multiple Hypothesis techniques seem promising to handle the desired factors, and we extend it from a normal tracker to serve as an analysis system.

2. RADAR BACKGROUND

The radar system of this study is a mechanically scanned search radar, with a rotating antenna that covers the entire search volume after one rotation. Observations (also known as *hits*) are received at regular intervals (typically 4 - 10 seconds), and from this a *tracker* creates *tracks* that represent estimated aircraft motion. A functioning radar device of a local company is used as subject for further simulations, and this radar has a search volume range of 65 km to a height of 8 km. It cannot make height detections, so only azimuthal and range measurements are therefore available.

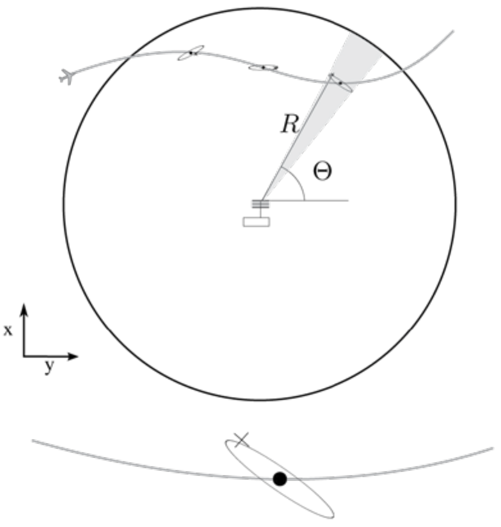


Figure 2: View from top of a radar situation, with detail of a noise covariance.

The radar observations are corrupted by noise as depicted in Figure 2, and the noise parameters are specified properties of a radar system (for our system, azimuth deviation 0.01222 radians and range deviation 18m). Visual sightings and position communication via radio can augment a tracking system. In combat situations Identification Friend or Foe (IFF) systems are used to identify aircraft, but usually only over specified zones. Radio silence during combat is however the standard. Thus even if an aircraft is identified at a specific moment, that certainty could be lost during close encounters with other targets.

3. TRACK MODELING

The radar observations received could be used as the approximated position of an aircraft, but there is more information available than this noise-corrupted data leading to better estimates. Two methods of tracking are discussed: Kalman filter and the Integrated Multiple Model technique.

3.1. *Kalman filter*

The Kalman filter [3] is a recursive filter with its gain being continuously adjusted based on the measurements received, the target dynamics and the noise models. The Kalman gain determines to what extent the estimate is either influenced by the measurement or influenced by the dynamic process model.

We used a linear Cartesian filter, with 4 or 6 states depending on whether acceleration is included as part of the model. A 4-state filter will perform better on simple linear motion, while a 6-state will track a turn better.

Predict:

The prediction for the state at time k is made before the measurement is received, by multiplying the state transition matrix A with the previous estimate. The covariance of the estimated state error P is predicted in a similar way, with the process noise covariance Q_k taking into account inaccuracies of the dynamic model

$$\begin{aligned}\hat{x}(k \mid k-1) &= A\hat{x}(k-1 \mid k-1)\\ P(k \mid k-1) &= AP(k-1 \mid k-1)A^T + Q_k. \end{aligned} \quad (1)$$

The choice of Q covariance is an important matter since it expresses the dynamics of the aircraft. A Kalman filter with a large covariance will track difficult manoeuvres better, but with estimation performance dropping.

Update:

The innovation ϵ is the difference between predicted measurement and the actual measurement

$$\epsilon(k) = z(k) - Hx(k \mid k-1). \quad (2)$$

The innovation covariance $S(k)$ of the estimated measurement includes the measurement noise covariance R_k, and this is used in the Kalman gain $K(k)$

$$S(k) = HP(k \mid k-1)H^T + R_k \quad (3)$$

$$K(k) = P(k \mid k-1)H^T S(k)^{-1}. \quad (4)$$

Now the state estimate $\hat{x}(k)$ is calculated by taking the state prediction and adjusting it according to the innovation and the Kalman gain. The state error covariance estimate is calculated for time step k with the use of the Kalman gain

$$\begin{aligned}\hat{x}(k) &= \hat{x}(k \mid k-1) + K(k)\epsilon(k)\\ P(k \mid k) &= (I - K(k)H)P(k \mid k-1). \end{aligned} \quad (5)$$

3.2. *Interacting Multiple Model filter*

The Interacting Multiple Model (IMM) estimator (as described by [4, p 455]) mixes the estimates from r Kalman filters according to how well it tracks the object. A Markov model describes the transition between the filter modes, meaning that there are specific probabilities that a target will change from one manoeuvre configuration to another.

In this way a low manoeuvre Kalman filter can be used for straight sections, while a high manoeuvre Kalman filter can be used for sections with sudden direction changes. As one performs better, its influence is increased in the mixed output, and similarly decreased as the performance drops.

Calculation of mix probabilities:

This step calculates the probability that one mode switched to another. The variable $\mu_{i|j}$ expresses the probability that mode M_i was active at $k-1$ given that M_j is active at k

$$\mu_{i\ |j}(k-1 \mid k-1) = \frac{p_{ij}\mu_i(k-1)}{\sum_{i=1}^{r} p_{ij}\mu_i(k-1)} = \frac{p_{ij}\mu_i(k-1)}{\bar{c}_j}. \quad (6)$$

Mixing:

Each filter calculates a new state by mixing all the filters together according to the mode transition probabilities, where r is the number of modes and $j = 1 \ldots r$

$$\hat{x}^{0j}(k-1 \mid k-1) = \sum_{i=1}^{r} \hat{x}^i(k-1 \mid k-1) \mu_{i|j}(k-1 \mid k-1). \quad (7)$$

The covariance is combined in a corresponding manner

$$\begin{aligned} &P^{0j}(k-1 \mid k-1)\\ = &\sum_{i=1}^{r} \mu_{i|j}(k-1 \mid k-1)\Big\{P(k-1 \mid k-1) +\\ &\left[\hat{x}^i(k-1 \mid k-1) - \hat{x}^{0j}(k-1 \mid k-1)\right] \cdot\\ &\left[\hat{x}^i(k-1 \mid k-1) - \hat{x}^{0j}(k-1 \mid k-1)\right]'\Big\}. \end{aligned} \quad (8)$$

Filter update and mode probability calculation:

With $\hat{x}^{0j}$ and P^{0j} assigned as mixed states of filter j, measurement $z(k)$ now updates each individual filter estimates in using these mixed states.

The likelihood Λ_j associated with the filter j is calculated with use of the innovation covariance S^{0j}, and assumed to be Gaussian with mean at the state position estimate $\hat{z}^{0j}$. Each filter has a mode probability μ, that represents the probability that the current filter is active given the measurement history. With $\bar{c}_j$ given by Equation 6, the likelihood and probability are given by

$$\begin{aligned}\Lambda_j(k) &= N[z(k); \hat{z}^{0j}, S^{0j}]\\ \mu_j(k) &= \frac{\Lambda_j(k)\bar{c}_j}{\sum_{j=1}^{r} \Lambda_j(k)\bar{c}_j}. \end{aligned} \quad (9)$$

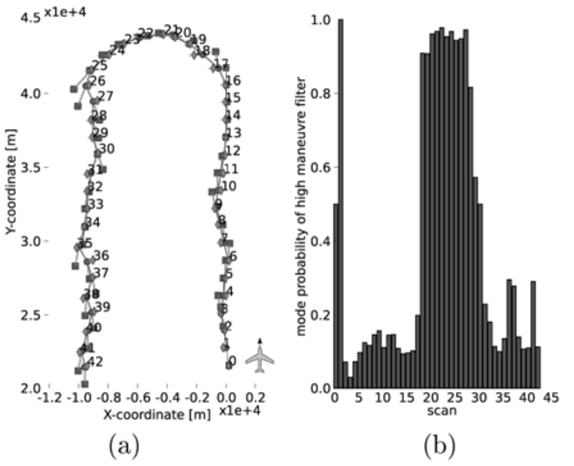


Figure 3: a) IMM tracking a $2g$ turn.
b) Mode probability of the high manoeuvre filter.

The mode probability for time k is calculated using the likelihood derived during the update step, together with the Markov transition probabilities and the mode probabilities from time $k-1$.

Estimate and covariance output:

Up to now each filter is mixed separately, and only influences each other during the mixing state. An output can be determined at any time by mixing these filters using the mode probabilities as weights. So it is similar to the mixing step in Equations 7 and 8, but this output is not fed back into the algorithmic loop

$$\hat{x}^j(k \mid k) = \sum_{j=1}^{r} \hat{x}^j(k \mid k)\mu_j(k \mid k) \tag{10}$$

$$\begin{aligned} P(k \mid k) \quad = \quad & \sum_{j=1}^{r} \mu_j(k) \Big\{ P^j(k \mid k) + \\ & \big[\hat{x}^j(k \mid k) - \hat{x}(k \mid k)\big] \cdot \\ & \big[\hat{x}^j(k \mid k) - \hat{x}(k \mid k)\big]' \Big\}. \end{aligned} \tag{11}$$

IMM Example:

Two filters form part of the ensemble: a 4-state low process noise filter to handle straight predictable path sections and a high process noise 6-state filter for tracking more intensive manoeuvres.

In this example an airplane flies at an altitude of $4km$ with a velocity of $300m.s^{-1}$ and then executes a $2g$ turn. Figure 3a shows the route measurements tracked with predictions and estimates, and Figure 3b the mode probability of the high manoeuvre filter on the right.

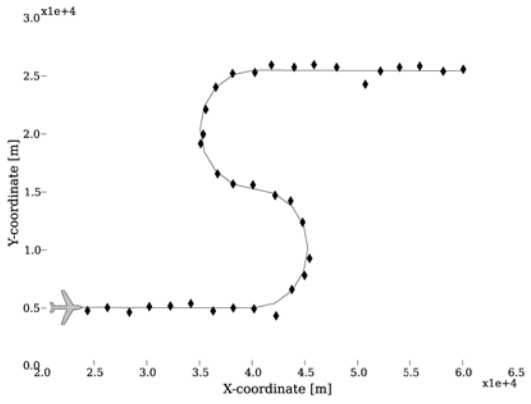


Figure 4: Test flight configuration.

Filter name	RMS(Δy)	Improvement
Kalman-4	500.602m	9.09%
Kalman-6	516.678m	6.17%
IMM	486.968m	11.56%

Table I: Comparison of filters.

3.3. *Comparison*

The performance of the Kalman and Multiple Model filters is now evaluated by comparing the root mean square error of the estimated position. At time step i the root mean square of difference between the true position y_i and the position estimate $\hat{y}_i$ is taken given by Equation 12.

$$RMS(\Delta y) = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2} \tag{12}$$

Figure 4 is an example of a flight configuration that contains straight sections and two tight $5g$ turns. The results are given in Table I, the improvement listed is the percentage it improves from the RMS error of the raw observation. The Kalman 6-state generally performs better on turns, but does not perform well in straight sections. The IMM filter mixes between a low process noise 4-state and a high-process noise 6-state filter, and in this case it performs the best. For other test configurations it also scores consistently higher, and has superior capability to cope with different kinds of motion.

4. TRACK MANAGEMENT

An estimation filter alone would be sufficient for the tracking of a single object under ideal conditions, but the noisy nature of the measurements can make the correct association a difficult task when multiple aircraft interact closely. This is made even more difficult when considering the additional challenges faced by a radar system such as false and missing reports, new targets and targets that end.

4.1. *Multiple Hypothesis Tracker*

The Multiple Hypothesis system manages a collection of hypotheses, each hypothesis representing a situation of possible targets and paths that could have been taken. So each hypothesis consists of a collection of tracks, where a track is a sequence of measurements and missed detections that represent the possible movement of an aircraft.

For each scan of measurements, the MHT looks at each of the existing hypotheses and creates new hypotheses for every possibility of track-measurement association. Missed detections, false targets and new targets are handled as well.

The next step is to estimate and predict the tracks. Each track of a hypothesis is represented by an estimation filter (Kalman or otherwise), and each one is updated according to the previously associated measurement.

Now the probability of a hypothesis is updated according to the measurement association and its nature. After this the hypotheses are compared, and those that are less likely are removed. At this moment the system awaits the next batch of measurements to restart the cycle.

4.2. *Theory*

Following Blackman [5], the probability that hypothesis K happened is given by:

$$\begin{aligned} Q_K \quad = \quad & C\beta_{FT}^{n_{FK}}\beta_{NT}^{n_K}\prod_{i=1}^{n_K}\Big[P_{TL}(D_i)P_D^{N_i} \\ & (1-P_D)^{D_i-N_i}\prod_{j=1}^{N_i}f(z_i(j))\Big] \qquad (13) \end{aligned}$$

where the algorithm is described below.

- $\beta_{FT}^{n_{FK}}$, $\beta_{NT}^{n_K}$: The sources of the tracks. An assumptions is made that targets arise randomly in space with uniform probability densities. β_{FT} represents the density for false targets, and β_{NT} for new targets. These densities are compounded for the false targets n_{FK} and the true targets n_K.
- $P_{TL}(D_i)$: The likelihood of a track disappearing from the search volume given the track length D_i.
- $P_D^{N_i}$, $(1-P_D)^{D_i-N_i}$: The probability of detection P_D (an aspect specific for a radar system) is compounded for the N_i detections of the track and the remainder $(1-P_D)$ for the $D_i - N_i$ missed detections.
- $f(z_i(j))$: The main contribution to the probability of a hypothesis is to what extent the observation $z_i(j)$ associate successfully with the track i. Finding this probability we use a multivariate Gaussian distribution to describe the probability density function of the residual error (the difference between the predicted and the real measurements).

The innovation covariance S and the observation prediction $\hat{z}$ of the track's Kalman filter is given by:

$$\begin{aligned} S \quad &= \quad HPH^T + R_c \\ \hat{z}(k \mid k-1) \quad &= \quad H\hat{x}(k \mid k-1). \end{aligned} \qquad (14)$$

The difference between the predicted and real observation is given by:

$$\tilde{z}(k) = z(k) - \hat{z}(k \mid k-1). \qquad (15)$$

The density function is evaluated with the observation as input:

$$f(z) = N[z(k); \hat{z}(k|k-1), S] = \frac{\mathrm{e}^{-\tilde{z}^T S^{-1} \tilde{z}/2}}{(2\pi)^{D/2}\sqrt{|S|}}. \qquad (16)$$

4.3. *Example*

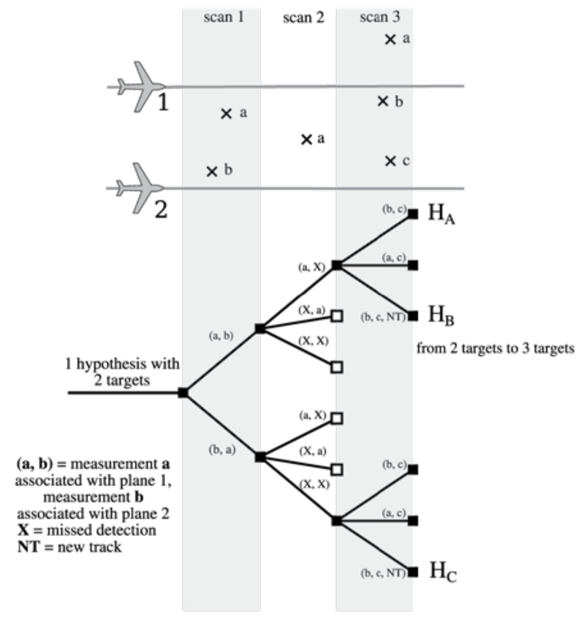


Figure 5: Example of hypothesis branching over time.

$$\begin{aligned} P(H_A) &= \beta_{nt}\beta_{ft}P_D^5(1-P_D)\prod g_{A1..A5} \\ P(H_B) &= \beta_{nt}^2 P_D^5(1-P_D)\prod g_{B1..B5} \\ P(H_C) &= \beta_{nt}^2\beta_f P_D^4(1-P_D)^2\prod g_{C1..C4} \end{aligned} \tag{17}$$

Figure 5 illustrates the tracking of the flight of two aircraft over three sets of received measurements and Equation 17 the resulting probabilities. A single hypothesis at the start represents two airplane tracks, and at each scan the hypothesis is branched into new hypotheses. In the figure, a branching tuple (a, b) indicates that measurement a of the current scan associated with track of plane 1 and measurement b associated with track of plane 2. H_A considers observation a of scan 3 correctly as a false target, while H_B and H_C consider it the start of a new target. During scan 2 there is only one observation so a target has been missed. H_C considers that single observation as a false target.

Figure 5 is a relatively simple example, and the hypothesis tree generated is substantially more involved than the one depicted here. Culling is therefore essential to prohibit an unmanageable number of hypotheses as demonstrated at the end of scan 2.

5. IDENTITY CONFIDENCE

A tracker outputs a series of tracks each consisting of a set of associated observations, while the other remaining observations are considered false targets. The task of identity confidence estimation is to take a track, and by comparison with the remaining observations, determine the probability that its identity integrity remained preserved.

The idea is to use the MHT algorithm retrospectively and to consider all the other likely possibilities. Given the initial tracks and their endings, the MHT can reconstruct possible track associations and combine the probability of all hypotheses sharing a specific track start and ending.

This is similar to using a MHT as a tracker, but differs by giving initial tracks as input. The algorithm can be applied with more focus on an area of interest, and since real time usage is not that important in this context, it can be simulated at greater depth. In essence, the hypotheses with all the different possible variations of the initial track are compared with hypotheses where the initial track do not occur. Where $H[tracks]$ select the hypotheses containing any of the supplied tracks, and $track[z_a, z_b]$ selects the tracks starting with z_a and ending with z_b,

$$P(z_a \to z_b) = \sum P(H[track[z_a, z_b]]) \tag{18}$$

gives the probability that an aircraft moved from starting observation z_a to the suspected end observation z_b. Using the MHT in this way the algorithm can handle any radar site setup, flying configuration and manoeuvre. The simplest case of interacting aircraft is a crossing bypass flight. Two airplanes flying directly next to each other in the same direction offers no chance of track identity preservation. On the other hand, flying past each other in opposite directions no confusion should be possible. Figure 6 shows results of different bypass configurations for 100 runs each expressed in histogram format. The probability considered is the probability that the airplanes did indeed cross. With 10 degree crossing the choice between the two possibilities (as shown in Figure 1) is equally likely, while at 140 degrees confusion is considered unlikely. These results confirm intuition.

In combat situations a visual inspection is a common manoeuvre. When a aircraft of unknown identity enters the airspace, another aircraft is dispatched to identify the target. This involves close quarters manoeuvring as the inspection aircraft swoops in behind its quarry and there is good chance of target identity confusion during the manoeuvre.

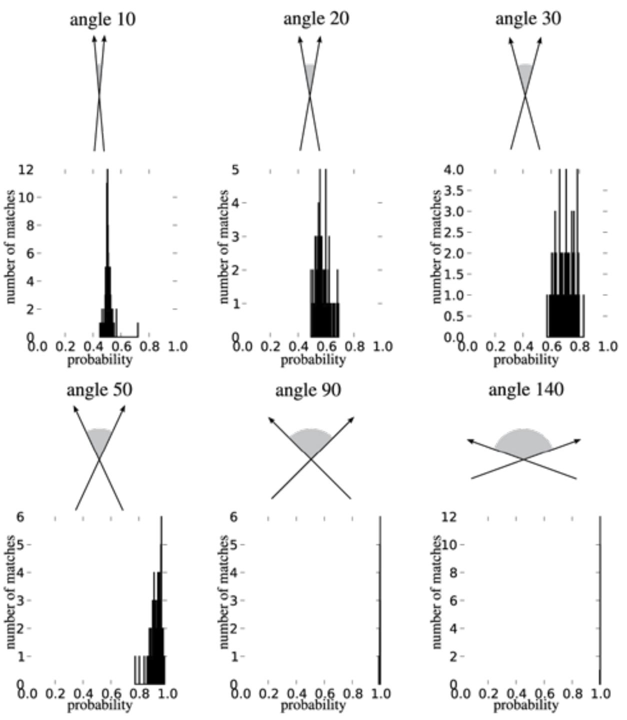


Figure 6: Identity confidence of a bypass flight.

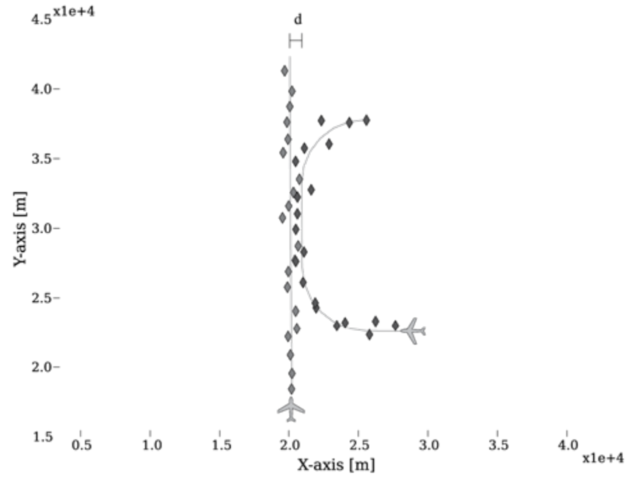


Figure 7: Inspection manoeuvre.

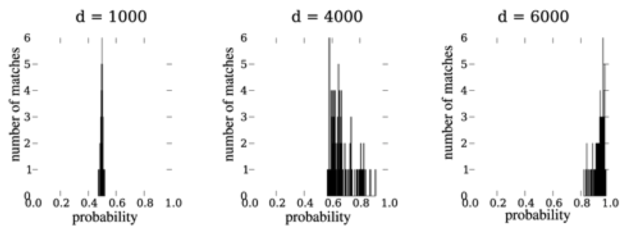


Figure 8: Identity confidences of inspection flights for various values of d.

This flight pattern is depicted in Figure 7. In this situation we will compare the probabilities obtained of paths bypasses various distances of d.

Figure 8 shows that at 1 km bypass there is not much certainty to be attached to any identity. For larger values of d the certainty rises until at the farthest bypass of 6 km the identities most probably remain preserved.

6. CONCLUSION

Posed with a problem from the industry to determine the identity confidence of radar tracking results, we have decided to approach the problem from the radar tracking methodology side. Multiple Hypothesis Tracking is a good way to extract probabilities from a scenario, and it uses numerous Kalman filters to estimate the best hypotheses. Variations of filtering were considered, and the Interacting Multiple Model filter proves to perform the best.

Extending the MHT and using it ex post facto on tracker output, we now have a robust system that can handle multiple aircraft while incorporating uncertainties of radar environment with ease. By applying the identity confidence system on simple bypass flight benchmark, the results obtained match the expected.

7. REFERENCES

[1] Y. Bar-Shalom and E. Tse, “Tracking in a cluttered environment with probabilistic data association,” *Automatica 11*, pp. 451–460, 1975.

[2] D. B. Reid, “An algorithm for tracking multiple targets,” *IEEE Transactions on Automatic Control*, vol. AC-24, no. 6, pp. 843–854, 1979.

[3] R. E. Kalman, “A new approach to linear filtering and prediction problems,” *Transaction of the ASME - Journal of Basic Engineering*, vol. 1960, pp. 35–45, 1960.

[4] Y. Bar-Shalom, X. R. Li, and T. Kirubarajan, *Estimation with Applications to Tracking and Navigation.* John Wiley and Sons, Inc, 2001.

[5] S. Blackman, *Multiple-Target Tracking with Radar Applications.* Dedham, MA: Artech House. Inc, 1986.

A NOTE ON DIFFERENCE SPECTRA FOR FAST EXTRACTION OF GLOBAL IMAGE INFORMATION

B.J van Wyk* M.A. van Wyk* and F. van den Bergh**

** French South African Technical Institute in Electronics (F'SATIE) at the Tshwane University of Technology, Private Bag X680, Pretoria 0001.*
*** Remote Sensing Research Group, Meraka Institute, CSIR, Meiring Naude Drive, Pretoria, South Africa.*

Abstract: The concept of an *Image Difference Spectrum,* a novel tool for the extraction of global image information, is introduced. It is shown that *Image Difference Spectra* are fast alternatives to granulometric curves, also referred to as pattern spectra. *Image Difference Spectra* are computationally easy to implement and are suitable for real-time applications.

Key words: Feature Extraction, Granulometries, Pattern Spectra

1. INTRODUCTION

Granulometries are useful tools for image analysis due to their ability to characterize size distributions and shapes and have been used extensively for feature extraction for classification, segmentation and texture analysis [1, 2]. Traditionally, granulometries are obtained using a series of openings or closings with convex structuring elements of increasing size. The granulometric analysis of an image results in a *signature* of the image with respect to the granulometry used which is referred to as granulometric curve or pattern spectrum. Due to the computational load associated with the calculation of granulometries, Vincent [3, 4], building on the work of Haralick *et al.* [2], proposed fast and efficient granulometric techniques using linear openings.

The *Image Difference Spectrum* algorithm proposed in this paper is not a morphological algorithm, but *similar* to morphological pattern spectra, the proposed algorithm extracts size distributions which can be used as global image features for a variety of pattern recognition applications.

2. IMAGE DIFFERENCE SPECTRA

The idea behind an *Image Difference Spectrum* is compactly summarised by the following two definitions:

Definition 1: An *Image Difference Spectrum*, Ω, is defined as a normalized representation of the number of occurrences of the lengths of *Segments of Increase* in each line of a greyscale image, Φ, having N rows indexed by n, and M columns indexed by m.

Definition 2: A *Segment of Increase* is a group of consecutive samples in a row of an image Φ, such that $\Phi(n,m+1)-\Phi(n,m)>\Phi(n,m)-\Phi(n,m-1)-\varepsilon$, where $\varepsilon \geq 0$ is a *Spectral Slack Parameter*.

Let Ω_i, $i=1,...,C$, be the *Image Difference Spectra* for C different classes. The *Spectral Slack Parameter,* ε, is a positive parameter chosen to maximize some norm between all Ω_i.

Unlike granulometric pattern spectra algorithms, the *Image Difference Spectrum* algorithm, given by the pseudo code in Section 3, is extremely easy to implement and has a linear complexity directly proportional to the number of pixels in the greyscale image.

3. PSEUDO CODE

```
initialize $k \leftarrow 0$, $\tilde{\Delta} \leftarrow 0$, $\Omega \leftarrow \mathbf{0}$
for $n = 1:N$
    for $m = 1:M$
        $\Delta \leftarrow \Phi(n,m) - \Phi(n,m-1)$
        if $\Delta > \tilde{\Delta} - \varepsilon$, increment $k$
        else, increment $\Omega(k)$ and set $k \leftarrow 0$
        end if
        $\tilde{\Delta} \leftarrow \Delta$
    end for
end for
scale $\Omega$ by dividing each element by the total number of pixels, i.e. $\frac{\Omega(k)}{NM} \forall k$.
```

4. EXPERIMENTAL RESULTS

The *Difference Spectrum* and Vincent's *Linear Greyscale Pattern Spectrum* [4] have been used to classify greyscale QuickBird satellite images over Soweto as formal suburbs or informal settlements. Since Vincent has demonstrated, using a variety of image applications, that *Linear Greyscale Pattern* spectra are faster and just as useful as conventional pattern spectra, only *Linear Greyscale Pattern Spectra* have been considered for comparison. The experimental results were obtained using MATLAB© code running on a 2 GHz Intel Core 2 Duo processor PC with 2GB RAM.

Only the first 10 bins of the spectra derived from the training and test sets, were used as the input to two feed forward neural networks, each having a single hidden layer with 6 neurons, trained using the Levenberg-Marquardt back-propagation algorithm. In fact, the *Image Difference Spectra* bins can be limited to only the first three without a degradation in performance.

One hundred images from selected Soweto suburbs, labelled by a built environment expert from the South African *Centre for Scientific and Industrial Research*, were equally divided into a training set and a test set. For all images *N=M=200.* Refer to Figures 1 and 2 for random image selections from the *formal suburb* and *informal settlement* training sets and their associated *Image Difference Spectra* and un-scaled *Linear Greyscale Pattern Spectra.*

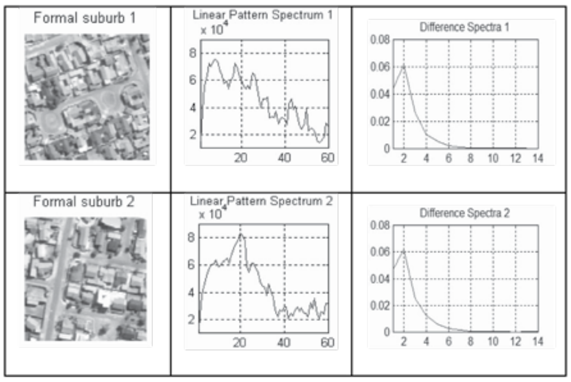


Figure 1: Soweto formal suburb images with their associated un-scaled *Linear Pattern Spectra* and scaled *Image Difference Spectra*.

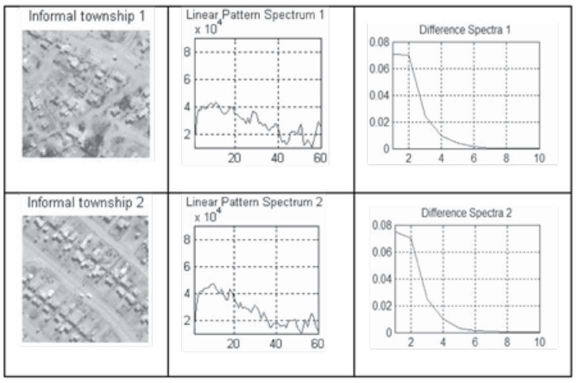


Figure 2: Soweto informal settlement images with their associated un-scaled *Linear Pattern Spectra* and scaled *Image Difference Spectra*.

For each of the two classes there were 25 training images and 25 test images. Using the Euclidean norm, a *Spectral Slack Parameter* of 3, was found experimentally to be optimal for the training set. For both algorithms a training and testing accuracy of 100% were achieved. The *Image Difference Spectrum* on average executed in 0.39 seconds and the *Linear Greyscale Pattern Spectra* on average in 1.11 seconds for a structuring element (which determines the number of bins) of 10. Calculating the *Linear Greyscale Pattern Spectra* using a structuring element of 60 took on average 14.15 seconds. Note that the number of bins for the *Image Difference Spectrum* are determined by the image characteristics and is not a parameter that can be selected.

5. CONCLUSION

A novel algorithm for the extraction of global image information was proposed and its application to the classification of images was presented. From the results obtained, it is clear that for the specific application considered, it performed well, both in terms of accuracy and computational speed. The algorithm has also been applied to the classification of seed mixture and steel surface images with equal success. The proposed algorithm can be used as a fast and robust alternative to granulometric pattern spectra.

ACKNOWLEDGMENTS

This work was financially supported by the Tshwane University of Technology (TUT) and the NRF under grant TRD2005070100036. Ignace T Toudjeu is thanked for implementing Luc Vincent's linear algorithm. The authors would also like to thank "Satellite image: DigitalGlobe" for assisting with the QuickBird image over Soweto.

6. REFERENCES

[1] P. Maragos, 'Pattern spectrum and multiscale shape representation', *IEEE Transactions on Pattern Analysis and Machine intelligence,* Vol. 11, No. 7, pp. 701-716, 1989.

[2] R. M. Haralick, S. Chen and T. Kanungo, 'Recursive opening transform', *in Proceedings of the IEEE International Computer Vision and Pattern Recognition Conference CVPR'92,* Champaign, IL, USA, pp. 560-565, 1991.

[3] L. Vincent, 'Granulometries and opening trees', *Fundamenta Informaticae*, Vol. 41, pp. 57-90, 2000.

[4] L. Vincent (1994) Fast grayscale granulometry algorithms, In Serra J. and Soille P. editors, *EURASIP Workshop ISMM'94: Mathematical Morphology and its Applications to Image Processing,* pp. 265-272, Fontainebleau, France, Kluwer Academic Publishers, 1994.

Notes