VISUAL INFORMATION PROCESSING
Proceedings of Pan-Sydney Area Workshop on Visual Information Processing
University of Sydney, Australia, 25th November, 1996
and

Technical Report 510
December 1996

Edited by:

Dr. David Dagan Feng
Coordinator, Biomedical & Multimedia Information Technology (BMIT) Group
Basser Department of Computer Science, University of Sydney, NSW, 2006

Prof. Hong Yan
Director, Imaging Science and Engineering Laboratory,
Department of Electrical Engineering, University of Sydney, NSW, 2006

Dr. Jesse Jin
Coordinator, Visual Information Processing Laboratory
School of Computer Science and Engineering, University of NSW, 2052

Organised and Sponsored by:
Basser Department of Computer Science, University of Sydney
Department of Electrical Engineering, University of Sydney
School of Computer Science and Engineering, University of NSW

BASSER DEPARTMENT OF COMPUTER SCIENCE
UNIVERSITY OF SYDNEY

Basser Department of Computer Science
University of Sydney
Sydney N.S.W. 2006
Australia
Preface

Recent advances in several key technological areas are leading to the large scale use of visual information. There is pressing requirement to efficiently capture, transform, analyse, manipulate, visualise, store, and retrieve images and pictorial data in wide spectrum of applications. Future information systems in commercial and scientific applications will have a high visual content, and it is necessary to address various aspects of visual information processing.

The workshop intended to bring a forum for people working in this area, and invited papers solicited on the following and related topics:

- Image Processing: gathering, coding, transforming, restoration, filtering, enhancement, space-variant processing
- Medical Imaging: feature extraction, segmentation, reconstruction, image analysis, telemedicine
- Multimedia: image and video indexing and retrieval, tele-conferencing, compression, CAD, authoring, semantic annotation, applications
- Visualisation: visual representation, modelling and reconstruction, scientific visualisation, interactive exploration, virtual reality
- Computer Graphics: computational geometry, surface rendering, volume rendering, modelling, animation, GUI and HCI
- Pattern Recognition: statistical pattern recognition and syntactic pattern recognition, pattern classification, neural networks, character recognition, document analysis
- Computer Vision: real-time imaging, robot vision and tracking, automatic inspection, remote control and sensing

The responses to this workshop and the Call for papers have been overwhelming. 80 people have attended the workshop. Due to tight time schedule and limited space for this one day workshop, the committee could only select 19 papers to give oral presentation from universities and industry. 18 of them have been included in this volume. We believe that this volume will be very useful for those who work in this area. Taking this opportunity, we would like to express our thanks to all of the contributors of this volume and the workshop. Without their enthusiasm and support, this proceedings and the workshop could not have been produced.

David Dagan Feng, Hong Yan and Jesse Jin

December, 1996.
# Table of Contents

<table>
<thead>
<tr>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dynamic Image Data Compression</td>
<td>1</td>
</tr>
<tr>
<td><em>Dino Ho and Dagon Feng</em></td>
<td></td>
</tr>
<tr>
<td>Recognition of Handwritten Characters Using Statistical Methods Based</td>
<td>8</td>
</tr>
<tr>
<td>on a Contour Following Algorithm</td>
<td></td>
</tr>
<tr>
<td><em>Dahai Cheng and Hong Yan</em></td>
<td></td>
</tr>
<tr>
<td>Real-Time Vision Processing for Autonomous Mobile Robots</td>
<td>22</td>
</tr>
<tr>
<td><em>Gordon Cheng and Alexander Zelinsky</em></td>
<td></td>
</tr>
<tr>
<td>Interaction Between Topological and Interval Relationships in Two</td>
<td>28</td>
</tr>
<tr>
<td>Dimensional Space</td>
<td></td>
</tr>
<tr>
<td><em>Nabil Mohammad</em></td>
<td></td>
</tr>
<tr>
<td>Dynamic Analyses of Recurrent Systems and Their Applications to</td>
<td>35</td>
</tr>
<tr>
<td>Pattern Recognition</td>
<td></td>
</tr>
<tr>
<td><em>Alan M. N. Fu and Hong Yan</em></td>
<td></td>
</tr>
<tr>
<td>Evaluating the Usability of Visual Formalisms</td>
<td>41</td>
</tr>
<tr>
<td><em>Ghassan Al-Qaimari</em></td>
<td></td>
</tr>
<tr>
<td>Comparative Study of Weighted Median Filters</td>
<td>49</td>
</tr>
<tr>
<td><em>William T. M. Dunsmuir and Yuan Zou</em></td>
<td></td>
</tr>
<tr>
<td>Automatic Human Face Detection and Recognition Under Non-uniform</td>
<td>56</td>
</tr>
<tr>
<td>Illumination</td>
<td></td>
</tr>
<tr>
<td><em>Toshiaki Kondo and Hong Yan</em></td>
<td></td>
</tr>
<tr>
<td>Real-Time Monitoring and Visualising the Performance of Parallel</td>
<td>62</td>
</tr>
<tr>
<td>Programs</td>
<td></td>
</tr>
<tr>
<td><em>Kei-Chun Li</em></td>
<td></td>
</tr>
<tr>
<td>Model Based Hybrid Coding of Human Face Images</td>
<td>70</td>
</tr>
<tr>
<td><em>Mustafa Sakalli and Hong Yan</em></td>
<td></td>
</tr>
<tr>
<td>Fixation Point Estimation and Transfer</td>
<td>78</td>
</tr>
<tr>
<td><em>Miroslav Trafkovic and Mark Hedley</em></td>
<td></td>
</tr>
<tr>
<td>Rotational Motion Artifact Correction in MRI</td>
<td>83</td>
</tr>
<tr>
<td><em>Chaminda Weerasinghe and Hong Yan</em></td>
<td></td>
</tr>
<tr>
<td>Application of Spiral Architecture to Edge Detection for Object</td>
<td>90</td>
</tr>
<tr>
<td>Recognition</td>
<td></td>
</tr>
<tr>
<td><em>Xiangjian He and Tom Hintz</em></td>
<td></td>
</tr>
<tr>
<td>Visualization of Data Dependency Analysis in Parallel Program Design</td>
<td>96</td>
</tr>
<tr>
<td><em>Elena Trichina, Bruce Thomas and Juha Oinonen</em></td>
<td></td>
</tr>
</tbody>
</table>
Visualising the Dynamics of Facial and Vocal Expression of Emotion
  E. C. Willock

Two Dimensional Wavelet Transforms
  Patrick M. Lenders and Anne M. Sjöström

An A-R Algorithm for Pattern Recognition
  D. B. Hoang and L. Esteban

“True Colour” Auto-Stereograms
  Allan Bromley and Barbara Karakassidis
Dynamic Image Data Compression

*Dino Ho and *Dagan Feng

*Biomedical and Multimedia Information Technology (BMIT) Group
Basser Department of Computer Science, The University of Sydney, NSW, Australia
e-mail: dinosaur@cs.us.oz.au and feng@cs.us.oz.au

Abstract—An algorithm for diagnostically lossless dynamic image data compression is proposed. The theory and implementation of this algorithm are presented. Taking advantage of domain specific knowledge related to medical imaging and the medical practice, we achieve very high compression ratios. An example using the fluoro-deoxy-glucose tracer and dynamic positron emission tomography is presented to evaluate the performance of the proposed algorithm. As a result of our study, the storage space for dynamic image data can be reduced by more than 95%, without loss in diagnostic quality.

Keywords—Image Compression, Positron Emission Tomography, Functional/Parametric Imaging.

I. INTRODUCTION

Computer technology is playing a more integral and indispensable role in modern medical and biomedical sciences. With widespread development in technology, digitization of the health-care industry has rapidly progressed to facilitate advance medical imaging and teleradiology. However, digitized medical images, even of moderate sizes, still require large volumes of data despite the advances in technology. This places considerable burden on storage, retrieval, processing and transmission of image data. With the recent introduction of dynamic medical imaging modalities, such as, positron emission tomography (PET), the proliferation of medical image data seems to be increasing at an exponential rate. For example, in a routine dynamic PET fluoro-deoxy-glucose (FDG) study, over 600 images would be required for a single patient study. Therefore, techniques that reduce these large volumes of medical image data without loss of useful information are of extreme interest.

Image compression is a popular technique used to reduce the amount of data to represent an image [1]. Depending on the compression scheme, algorithms can be divided into two main categories, lossless and lossy compression algorithms. Lossless compression algorithms allow for perfect reconstruction of the original images from compressed data. These algorithms yield modest compression ratios typically between 1.7:1 to 2.1:1 for medical image data [2]. To achieve higher compression ratios, lossy compression algorithms are required. Using lossy compression algorithms, the original images can only be reconstructed approximately from compressed data.

The challenges posed by medical imaging are the development of compression algorithms that are nearly lossless for diagnostic purposes, yet attain high compression ratios to reduce storage, retrieval, transmission, and processing. For dynamic image data, both spatial and temporal redundancies in the data need to be considered. In this paper, we focus our attention to diagnostically lossless compression of dynamic image data. The theory developed is generally applicable to all types of imaging modalities where dynamic image data is acquired. To illustrate the practicality of the proposed compression algorithm, an example utilizing simulated PET-FDG data is presented. The tracer FDG is an analog of glucose and is used to study metabolic rates of various tissue structures in the body. The study was restricted to simulated data, since, only with simulations are the original, error-free parametric images available for comparative purposes.

II. TRACER KINETIC MODELING AND PARAMETRIC IMAGING

Tracer kinetic techniques are widely used in PET to extract valuable information from dynamic processes in the body. This information is usually defined in terms of a mathematical model \( \mu(t|p) \) (where \( t = 1, 2, \ldots, T \) and \( p \) are the model parameters), whose parameters describe the delivery, transport and biochemical transformation of the tracer. The driving function for the model is the plasma blood input function, which is often obtained from blood sampling [3]. Measurements acquired by PET define the tissue time activity curve (TAC), or output function, denoted \( z_i(t) \), where \( t = 1, 2, \ldots, T \) are discrete sampling times of the measurements, and \( i = 1, 2, \ldots, I \) corresponds to the \( i \)-th pixel in the imaging region. The purpose of functional / parametric imaging is to obtain tracer TACs and parameter estimates \( \hat{p} \) for each pixel in the imaging region. These parameters may provide information of interest in themselves, or may be used to define other physiological parameters, such as the metabolic rate of glucose (MRGl) [3].

Fig.1 shows the conventional steps involved in parametric image generation from the complete set of acquired PET projection data. Once the projection data is reconstructed, parameter estimation is performed on a pixel-by-pixel basis using certain rapid estimation algorithms [3], [4], [5]. In this paper, we use the traditional weighted non-linear least
III. COMPRESSION ALGORITHM

The goal of a compression algorithm is to reduce the size of data required for storage, retrieval and transmission, and facilitate in analysis of the compressed data. To achieve this, efficient methods are required to exploit and code redundancies in the data. Fig. 2 gives an overview of the stages involved in the proposed compression algorithm.

- **Stage 1**: Exploit temporal redundancies in the data. Applying the image optimal sampling schedule (OSS) design developed by the Biomedical and Multimedia Information Technology (BMIT) Group [5], [6], the number of temporal frames can be reduced while preserving data quality and fidelity (Section IIIA).

- **Stage 2**: Exploit spatial redundancies in the data. Using cluster analysis, the reduced set of temporal frames can be further compressed to a single indexed image. This indexed image contains a mapping of the cluster groups for the reduced set of temporal frames and corresponds to the spatial information in the PET data. The respective temporal information for each cluster group is contained in an index table. This table is sequentially indexed by the cluster group and each index contains the mean TAC cluster values for that group (Section IIIB).

- **Stage 3**: Efficient coding of image data. Applying standard still-image compression techniques to the single indexed image, the dynamic data was further compressed (Section IIIIC).

For a detailed discussion of OSS and still-image compression techniques we refer you to [6], [5] and [7] respectively. For the purpose of brevity, only the essential concepts are summarized below.

A. *Stage 1: Compression in the temporal domain*

In PET studies, the reliability of the temporal frames is directly influenced by the sampling schedules and durations used to acquire the data. The longer the durations and greater the radio-activity counts, the more reliable the temporal frames. However, in order to obtain quantitative information from the dynamic processes, a certain number of temporal frames are required. Recently, the BMIT Group has shown that the minimum number of temporal frames required is equal to the number of model parameters to be estimated [5]. Based on this, an algorithm that automatically determines...
OSS and maximizes the information content of the acquired PET data was developed [6]. The developed algorithm utilizes the accumulated / integral PET measurements.

In this paper, we apply BMIT-OSS to acquisition of PET projection data. This reduces the number of temporal frames obtained and therefore, reduces data storage. Furthermore, as fewer temporal frames are reconstructed the computational burden posed by image reconstruction is reduced. For a more detailed discussion on the theory and principles of BMIT-OSS we refer you to [5] and [6].

B. Stage 2: Compression in the spatial domain

Cluster analysis aims at grouping and classifying pixel-wise TACs, $z_i(t)$ (where $i = 1, 2, \cdots, I$) into $C_i$ cluster groups (where $j = 1, 2, \cdots, J$) by natural association according to self-similarity (or dissimilarity) characteristics. Many algorithms have been developed and used for clustering [8], [9]. They can be divided into direct (constructive) or indirect (optimization) algorithms depending on whether a criterion measure is used during cluster analysis. The direct algorithms perform clustering without the necessity of a criterion measure, whereas indirect algorithms use the criterion measure to optimize clustering. Clustering algorithms can be further classified as agglomerative or divisive, according to whether classification is in a top-down or bottom-up direction. With agglomerative clustering, TACs are coalesce to the nearest cluster groups according to a threshold in a bottom-up direction. Agglomerative algorithms are more versatile than divisive algorithms as they can be used with both qualitative and quantitative data.

In this paper we use an indirect agglomerative clustering algorithm based on the traditional Euclidean distance criterion measure [8]. Although this algorithm is simple compared to other more sophisticated algorithms, excellent classification of cluster groups are obtained. However, in general, the application of this algorithm to clinical data does not provide optimal classification of cluster groups. This is due to the inherent noise in clinical data, and the fact that TAC values over the whole image volume do not take a number of discrete values but rather a continuum of values [9]. In this case, an interesting analogy with principal component and factor analysis may be made where similar considerations apply [10], [11].

Using the results of cluster analysis, an index table containing the mean TAC within each cluster group $\bar{z}_{ij}(t)$ and an indexed image can be formed. The indexed image represents a mapping of the cluster groups to their respective pixel TAC locations. This image together with the index table form the basis of the compressed temporal / spatial data. With PET, the number of distinguishable cluster groups will generally not exceed 64. This means that a 8-bit indexed image is sufficient to represent the cluster mapping.

C. Stage 3: Image Compression

Image compression addresses the problem of reducing the amount of data required to represent an image. A lossless compression scheme is considered in this paper for further compression of the indexed image. In general, the underlying basis of lossless compression is efficient coding of image data using as few bits as possible. Various coding techniques have been proposed for lossless compression [1]. In general, these techniques use predictive models or multi-resolution image models to reduce redundancies, and encode residuals using optimal encoders, such as, Huffman, Lempel-Ziv or arithmetic coding [4].

In this paper, we compress and store the indexed image obtained from cluster analysis using the Portable Network Graphics (PNG) file format [12]. The coding technique presently defined and implemented for PNG is based on deflate/inflate compression with a 32K sliding window. Deflate compression is based on a LZ77 derivate, and encoded using fixed or custom Huffman codes. The PNG file format was chosen over other lossless image compression file formats due to its portability, flexibility and it being legally unencumbered. For a more detailed discussion of the PNG file format and its included features, we refer you to [12].

IV. COMPRESSED DATA PARAMETRIC IMAGING APPROACH

This approach uses the compressed data obtained from the proposed compression algorithm for tracer kinetic modeling, and subsequent generation of parametric images. In Fig.3 the various steps involved in generating parametric images from the compressed data are shown.

- **Step 1**: De-compression of indexed image. Since lossless compression is used for compressing the indexed image, a perfect reconstruction of the image is possible.

- **Step 2**: Tracer kinetic modeling and parameter estimation. Using the cluster TACs defined in the index table, obtain parameter estimates for the tracer kinetic model by fitting $\xi(t)$ to the model equation $\mu(t|p)$. Subsequently, calculate the physiological parameter of interest using the obtained estimates.

- **Step 3**: Pixel-wise mapping. Map the obtained estimates and calculated physiological parameters for each cluster TAC to their respective pixel locations by referencing the indexed image. The resultant images correspond to the generated parametric images.
where $\mathbf{p}$ is the unknown vector of parameters to be estimated, $\mathbf{p} = (k_1^*, k_2^*, k_3^*, k_4^*)$. $k_1^* - k_4^*$ are the rate constant parameters of the FDG model. $c^*_p(t)$, $c^*_m(t)$ and $c^*_t(t)$ correspond to the concentration of FDG in plasma, FDG in tissue and phosphorylated FDG in tissue respectively. Once the unknown model parameters have been estimated, the physiological parameter for the metabolic rate of glucose (MRGlc) can be calculated as

$$MRGlc = \frac{1}{LC} \frac{k_1^* k_3^*}{k_2^* + k_3^*} c_p$$

(2)

where LC (= 0.418) denotes the lumped constant that summarizes the difference between FDG and glucose in transportation and phosphorylation, and $c_p (= 91.9mg/100ml)$ corresponds to the “cold” glucose concentration in plasma [3]. The parameter values for various regions in the simulated phantom are obtained from real PET studies and correspond to tissue variations of grey, white and whole matter [3]. $c_p^*$, the concentration of FDG in plasma was obtained from human studies [4]. The sampling durations for the measurement of the PET data using the conventional approach are consistent with the conventional sampling schedule (CSS) for routine FDG studies. The OSS proposed by Li et al. [6] was used for generating projection data for the compressed data approach. The sampling times for CSS are given by 0.2, 0.4, 0.6, 0.8, 1.0, 1.2, 1.4, 1.6, 1.8, 2.0, 2.5, 3.0, 4.0, 5.0, 6.5, 10.0, 15.0, 20.0, 30.0, 60.0, 90.0 and 120.0 min, and for OSS by 2.733, 15.683, 77.066 and 120.0 min [6].

To simulate PET measurement noise, an additive noise contribution following a Poisson distribution was imposed on the simulated TACs measurements using CSS. Noise corresponding to a 20% 90% deviation in the first 10 PET measurements (represents a 2% deviation in the last PET measurement) was applied. This noise level is comparable to that expected in real clinical data. For image reconstruction, the popular filtered back-projection algorithm using the Generalized Hamming filter (Hamming distance $\alpha = 0.5$) was used [13].

### A. Compression Measures

Various measures have been used for reporting and comparing compression results [1]. The most commonly used measures include the compression ratio (CR = $t/l$, where $t$ is the original file size and $l$ is the compressed file size; eg. 2.0:1) and relative compression (RC = $(t - l)/t$; eg. 50.0%). In this paper, we use these measures and include an additional measure, compression gain (CG = $10 \log (t/l)$; eg. 60.3 dB) [1]. CG has several additional advantages.
over the commonly used measures (CR and RC). First, this measure does not express compression relative to the original file and therefore can be used as a measure with respect to any standard. Second, CG is additive and allows us to simply add together effects of multiple cascaded compressors. For a more detailed discussion on CG we refer you to [1].

B. Image Quality Measures

For any lossy compression algorithm, a measure is required to evaluate the image quality and fidelity after de-compression of the compressed data. In the case of PET, we require parametric images generated using the compressed data to have equivalent qualitative and quantitative quality as images obtained from the complete set of PET data. In this paper, the root mean square error (RMSE) and the peak signal to noise ratio (PSNR) measures are considered. RMSE is evaluated as

\[
RMSE = \sqrt{\frac{1}{I} \sum_{i=1}^{I} (f_i - \hat{f}_i)^2} 
\]

where \( I \) corresponds to the number of pixel in the image, \( f_i \) is the value of the \( i \)-th pixel in the original / true parametric image and \( \hat{f}_i \) is the value of the \( i \)-th pixel in the parametric image based on the image analysis of the complete or compressed data. The PSNR is a measure of image fidelity, its units are in decibels (dB). It is defined as

\[
PSNR = 20 \log_{10} \left( \frac{RMSE}{2^n - 1} \right) 
\]

where \( n \) is the number of bits per pixel in the image.

VI. Result / Discussion

In Table I, results for each stage of the compression algorithm for a typical PET study are shown. The proposed algorithm achieved an overall CR of 82.8:1, this CR corresponds to a CG of 441.67 and a RC of 98.8%. Values for MRGlkh, \( k_1^{\ast} \), \( k_2^{\ast} \), \( k_3^{\ast} \) and \( k_4^{\ast} \) obtained from parametric images generated using the complete and compressed dynamic image data are shown in Table VI. The determined values using the compressed data were in excellent agreement with the original, error-free parameter values.

<table>
<thead>
<tr>
<th>Stage 1: Compression in the temporal domain</th>
<th>Stage 2: Compression in the spatial domain</th>
</tr>
</thead>
<tbody>
<tr>
<td>Optimal sampling schedule (4 time frames OSS)</td>
<td>Cluster analysis</td>
</tr>
<tr>
<td>( 256 \times 256 \times 4 \times 2 )</td>
<td>Assume a maximum of 256 different clusters (based on the 8-bit indexed image)</td>
</tr>
<tr>
<td>( = 524,288 ) bytes</td>
<td></td>
</tr>
<tr>
<td>CR = 5.5:1, CG = 170.47 and RC = 81.8%</td>
<td>i) indexed image</td>
</tr>
<tr>
<td></td>
<td>( 256 \times 256 \times 1 )</td>
</tr>
<tr>
<td></td>
<td>( = 65,536 ) bytes</td>
</tr>
<tr>
<td></td>
<td>ii) cluster index</td>
</tr>
<tr>
<td></td>
<td>( 256 \times 4 \times 2 )</td>
</tr>
<tr>
<td></td>
<td>( = 2048 ) bytes</td>
</tr>
<tr>
<td></td>
<td>Total size = 67,584 bytes</td>
</tr>
<tr>
<td>CR = 7.8:1, CG = 204.87 and RC = 87.1%</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Stage 3: Still image compression</td>
</tr>
<tr>
<td>PNG lossless still-image compression</td>
<td>Assume CR = 2.0:1, typical for medical images</td>
</tr>
<tr>
<td></td>
<td>i) indexed image</td>
</tr>
<tr>
<td></td>
<td>( = 32,768 ) bytes</td>
</tr>
<tr>
<td></td>
<td>ii) cluster index</td>
</tr>
<tr>
<td></td>
<td>( = 2048 ) bytes</td>
</tr>
<tr>
<td></td>
<td>Total size = 34,816 bytes</td>
</tr>
<tr>
<td>CR = 1.9:1, CG = 66.33 and RC = 48.3%</td>
<td>Overall:</td>
</tr>
<tr>
<td></td>
<td>CR = 82.8:1, CG = 441.67 and RC = 98.8%</td>
</tr>
</tbody>
</table>

that parametric images generated using the conventional approach are quite noisy. For the proposed approach, cluster analysis acts as an averaging filter and significantly reduces noise in the generated parametric images.

For generation of parametric images, the major computations involved are curve fittings of TACs. Since the number of TACs required to be fit using the compressed data are significantly smaller, the proposed parametric image approach is significantly faster than the conventional approach. If we neglect background pixels, over 31,160 curve fittings were required based on the conventional approach. However, with the proposed approach, the number of curve fittings required is determined by the number of TACs in the index table. In the worst case scenario 256 curve fittings are required based on the
Table II. Parametric image physiological parameter values

<table>
<thead>
<tr>
<th>Region</th>
<th>MRGlc</th>
<th>$k_1^*$</th>
<th>$k_2^*$</th>
<th>$k_3^*$</th>
<th>$k_4^*$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Background</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0000</td>
</tr>
<tr>
<td>Region 1 (Whole)</td>
<td>5.3032</td>
<td>0.0780</td>
<td>0.1195</td>
<td>0.0535</td>
<td>0.0063</td>
</tr>
<tr>
<td>Region 2 (White)</td>
<td>3.4692</td>
<td>0.0540</td>
<td>0.1090</td>
<td>0.0450</td>
<td>0.0058</td>
</tr>
<tr>
<td>Region 3 (Grey)</td>
<td>7.2415</td>
<td>0.1020</td>
<td>0.1300</td>
<td>0.0620</td>
<td>0.0068</td>
</tr>
</tbody>
</table>

Analysis of complete projection data

Conventional approach

<table>
<thead>
<tr>
<th>Region</th>
<th>MRGlc</th>
<th>$k_1^*$</th>
<th>$k_2^*$</th>
<th>$k_3^*$</th>
<th>$k_4^*$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Background</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0000</td>
</tr>
<tr>
<td>Region 1 (Whole)</td>
<td>5.2524</td>
<td>0.0813</td>
<td>0.1291</td>
<td>0.0536</td>
<td>0.0056</td>
</tr>
<tr>
<td></td>
<td>(stddev)</td>
<td>0.3246</td>
<td>0.0048</td>
<td>0.0013</td>
<td>0.0000</td>
</tr>
<tr>
<td>Region 2 (White)</td>
<td>3.5639</td>
<td>0.0548</td>
<td>0.1177</td>
<td>0.0453</td>
<td>0.0052</td>
</tr>
<tr>
<td></td>
<td>(stddev)</td>
<td>0.0590</td>
<td>0.0004</td>
<td>0.0013</td>
<td>0.0013</td>
</tr>
<tr>
<td>Region 3 (Grey)</td>
<td>7.2590</td>
<td>0.1083</td>
<td>0.1383</td>
<td>0.0607</td>
<td>0.0059</td>
</tr>
<tr>
<td></td>
<td>(stddev)</td>
<td>0.1476</td>
<td>0.0018</td>
<td>0.0010</td>
<td>0.0008</td>
</tr>
</tbody>
</table>

Analysis of compressed data

Proposed compressed data approach

<table>
<thead>
<tr>
<th>Region</th>
<th>MRGlc</th>
<th>$k_1^*$</th>
<th>$k_2^*$</th>
<th>$k_3^*$</th>
<th>$k_4^*$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Background</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0000</td>
<td>0.0000</td>
</tr>
<tr>
<td>Region 1 (Whole)</td>
<td>3.5611</td>
<td>0.0822</td>
<td>0.1298</td>
<td>0.0539</td>
<td>0.0056</td>
</tr>
<tr>
<td>Region 2 (White)</td>
<td>3.3710</td>
<td>0.0541</td>
<td>0.1224</td>
<td>0.0484</td>
<td>0.0054</td>
</tr>
<tr>
<td>Region 3 (Grey)</td>
<td>7.2508</td>
<td>0.1081</td>
<td>0.1391</td>
<td>0.0611</td>
<td>0.0060</td>
</tr>
</tbody>
</table>

In this paper, a new diagnostically lossless algorithm for dynamic image data compression was proposed and systematically evaluated. In order to fully exploit data redundancies in the acquired dynamic data, domain specific knowledge related to medical imaging and the medical practice were incorporated into the compression scheme. Redundancies in the data in both the temporal and spatial domain were exploited. The results showed that the proposed algorithm reduce required storage space by more than 95% without loss in diagnostic quality and greatly reduced the computational complexity for further image analysis, and generation of parametric images. Therefore, the proposed algorithm for dynamic image data compression is expected to be very useful in medical image data management and telecommunication.

VII. CONCLUSION

REFERENCES

Fig. 4 (a) Dynamic phantom used in the simulation study for generating dynamic PET projection data. The FDG model was used to define TAC's for various regions in the phantom. Rate constants corresponding to various tissue types are used, for grey matter, $k_1' = 0.1020$, $k_2' = 0.1300$, $k_3' = 0.0620$ and $k_4' = 0.0068$; for white matter, $k_1' = 0.0540$, $k_2' = 0.1090$, $k_3' = 0.0450$ and $k_4' = 0.0058$; and for whole matter, $k_1' = 0.7080$, $k_2' = 0.1195$, $k_3' = 0.0535$ and $k_4' = 0.0063$. (b) (f) Vertical profiles for the generated parametric images. The solid lines correspond to profiles obtained from the original, error-free parametric images, the dashed lines correspond to the conventional approach, and the dotted lines correspond to the proposed compressed data approach.
Recognition of Handwritten Characters Using Statistical Methods Based on a Contour Following Algorithm

Dahai Cheng and Hong Yan
Department of Electrical Engineering
University of Sydney, NSW 2006, Australia
E-mail: dcheng@ee.usyd.edu.au

Abstract. In this paper, a new statistical method is proposed for recognition of unconstrained handwritten characters based on a contour following algorithm. The whole contour following algorithm includes: Preprocessing, contour extraction, redundant pixel removal, contour following and post processing. This algorithm has two step memories, and it can remove some noisy pixels along contour automatically. In the recognition process, first, the characters are classified into three groups according to their topological properties. Then, the features of normalized length of character contour, the normalized area of character and the normalized distance function of outer contour are used for recognition of characters with one contour. After that, the relative positions of centroids of outer and interior contours, the position of interior centroid and the mean and the variance of the distance function of outer contour are used for recognition of characters with multiple contours. Finally, some special models are established and used for recognition of broken characters. In our experiment, 1000 characters from the NIST database are used for training and 5073 unseen characters are used for testing. The recognition rate has reached 98.94% with a reliability of 99.78%, a substitute rate of 0.21% and a rejection rate of 0.85%.

1 Introduction

Recognition of handwritten characters has been a popular topic of research for many years [1]-[5]. Many approaches have been proposed to solve this problem. These methods can be classified into two classes. In the first class, the structure information, such as the features of strokes, junctions, arcs, concavities and convexities and endpoints, are used for recognition [6]. Most syntactical classification techniques belong to this class. In the second class, the statistical information, such as edge features and measurements of moments, are used for recognition.

In this paper, we proposed a statistical recognition method based on various contour information. In our method, all features used in statistical analysis from contour information. In order to extract these features, a contour following algorithm [11] is used to trace character contours and a tree classifier [7]-[10] is employed for recognition. First, the character contours are followed from outside to inside (or from top to bottom). Then, the characters are classified into three groups according to their topological properties. Finally, various contour information are used for further recognition.

There are five sections in this paper: Introduction, Character preprocessing, Recognition of handwritten characters, Experiment results and Conclusion.

In character preprocessing, the first step is contour smoothing, which includes small hole and gap filling, broken character mending etc. Then, the character contours are followed one by one. In this paper, a new algorithm named “Two step memory algorithm” (TSMA) is proposed for contour following. In order
to obtain the similarity function for further recognition, the normalized distance function is used to represent the shapes of character.

In the recognition process, the first step is feature extraction. The features we used include: the normalized distance function of character contours, the mean and the variance of distance function (the distance from the centroid to the pixel on contour), the normalized length of contour, the normalized area of characters. The normalized distance function is calculated based on centroid of outer contour the starting point. The mean and the variance of distance function are calculated from outer character contour and normalized by its longest distance from centroid to contour. The normalized length of contour and the normalized area of character are calculated no matter whether the character is broken or not. For a broken character, the contour length is equal to the sum of the length of two (or more) contours, and the area is equal to the sum of the areas of all components in the character. With these features, some broken characters can still be recognized correctly.

The recognition process the characters are divided into three groups. In group one, every character has only one contour. The characters in this group are recognized by a shape recognition method based on their normalized distance function. In group two, every character has two contours. In this group, "0", "6", "9", "2" and "5" with two contours are recognized by the features of the mean and the variance of distance function of contour, the position of interior centroid, the relative position of centroids of outer and interior contours, the topological relationship between the first and second contour and the normalized distance function of outer contour of character. In group three, basically there is only one character "8". Actually, because of complex of writing styles, some unusual characters with three contours are still in this group. These characters are discriminated by the comparison of contour shapes of models and input character.

2 Character preprocessing and contour following algorithms

2.1 Character contour smoothing

The problem of handwritten character recognition is very complicated. Each character has many different writing styles and a character can be broken. To improve recognition performance, broken characters should be mended. Sometimes there are some holes or gaps on a contour. In this case, the holes or gaps should be filled before contour extraction. Examples of characters before and after mending and hole filling are shown in Fig. 1.

2.2 Contour following algorithm

In this paper, a new contour following algorithm named the Two Step Memory Algorithm is proposed to follow contours of characters. The whole contour following process includes: prefollowing processing, contour extraction, redundant pixel removal, contour following and post following processing.

2.2.1 Prefollowing processing of characters before contour extraction

After character contour smoothing, it still needs processing. Because we hope after contour extraction, there is no connecting pixel between the outer contour and interior contours. In some cases, there are only one or two pixels on some part of character between outer and interior contours. In this case, it will be difficult to extract contours out, so we have to make this part thicker before the contour extraction.

The basic idea of making the contour thicker is described as follows. If a pixel is nonzero, it must be a pixel on the character. But, if the number of pixels in one of the four directions (shown in Fig. 2) is
smaller than 3, we will fill a nonzero pixel in the order which is shown in Fig. 2 until the thick between outer and interior contours is enough for extraction (That means after extraction, there is no connecting pixel between outer and interior contours).

Some characters before and after prefollowing processing are shown in Fig. 3.

**2.2.2 Contour extraction and redundant pixel removal**

After preprocessing, the contours of characters will be extracted. The contour extraction in this paper is very simple. If a pixel is non-zero and at least one of its four 4-connected pixels is zero, we believe this is contour pixel. Otherwise, it belongs to non-contour pixel. The characters of “8” before and after contour extraction are shown in Fig. 7.

After contour extraction, usually there are some redundant pixels on contour. Sometimes these redundant pixels will affect the contour following, so these redundant pixels should be removed. The redundant pixel removal process is based on the request that every contour pixel can only have two 8-connected pixels around it. If there is a pixel which has more than two 8-connected pixels around it, there must be one or even more redundant pixels around it, the redundant pixels will be detected and removed. The redundant pixel removal process is shown in Fig. 4. In Fig. 4(a), pixel 3 is a redundant pixel, it could be detected and removed.

**2.2.3 Contour following algorithm: Two Step Memory Algorithm**

After extraction of outer and interior contours, we follow every contour from outside to inside or from top to bottom. We use a new contour tracing algorithm, which has two step memory algorithm. This algorithm can avoid some disturbances along the contour, and it can also avoid some stops during tracing. The tracing process startings from the upper left pixel on the contour, searching for next nonzero pixel in a clockwise direction.

In some cases, this contour tracing process may stop. For example, for the searching process shown in Fig. 5, the current position is A. According to this method, the next point is B and from point B, the next point is C. But the searching process stops at C. In order to avoid this problem, the position B must be remembered. If the searching process has not finished and cannot continue, it has to go back to original position B, and remove C at the same time, and then the searching process startings again.
(x_p, y_p) as the reference point, where

\[ x_p = \frac{1}{n} \sum_{i=1}^{n} x_i \]  \hspace{1cm} (1)

\[ y_p = \frac{1}{n} \sum_{i=1}^{n} y_i \]  \hspace{1cm} (2)

where \((x_i, y_i)\) is the i-th pixel on the contour, \(n\) is the number of pixels on the contour. We choose the starting point which has the longest distance from the centroid.

In order to get the normalized distance function of the contour, we have to follow character contour from the starting point again. Because for different character, the position of starting point always different. If we want to get a unique distance function (that means we have to follow every character contours in same direction), we have to keep the following in clockwise direction. The contour following orders for different starting point positions are shown in Fig. 8.

If the starting point is at the upper-right of the centroid, in order to keep the contour following direction in clockwise direction, we have to choose the following contour tracing order: \((i, j + 1), (i + 1, j + 1), (i + 1, j), (i + 1, j - 1), (i, j - 1), (i - 1, j - 1), (i - 1, j), (i - 1, j + 1)\), which is shown in Fig. 8. If the starting points are at the upper-left lower-right and
lower-left of the centroid, the contour following orders are shown separately in Fig. 8.

3 Character recognition using statistical methods

3.1 Feature extraction

For the purpose of classification, we have to find some features which can represent properties of contours of different characters. The features used in this paper are normalized distance function of contour (the distance from centroid to pixel on contour), length of character contours, area of characters, centroids of outer contours and interior contours, starting point of outer contours. Methods for extraction of these features are described below.

3.1.1 The contour distance function from the centroid

The contour distance function from the centroid is a simple and effective way to describe a contour. In order to derive the contour distance function, we have to find a reference point, which is invariant to the rotation of the character. Here we use the centroid \((x_p, y_p)\) as the reference point, where

\[
x_p = \frac{1}{N} \sum_{i=1}^{N} x_i
\]

\[(3)\]

\((x_i, y_i)\) is a pixel on the contour, and \(N\) is the number of pixel on contour. We choose the starting point which is of longest distance from the centroid. When the point moves along the contour at clockwise direction, we get the contour distance function. For convenience of comparison, the contour distance function should be normalized. The normalized contour distance function is defined below

\[
y_p = \frac{1}{N} \sum_{i=1}^{N} y_i \tag{4}\]

\[
ND(n) = \frac{D(n)}{L_M} \tag{5}\]

where \(L_M\) is the longest distance from the centroid to contour. The normalized contour distance function of character "7" is shown in Fig. 9.

For the purpose of recognition, we have to extract some features from the normalized contour distance function. The features we will use are the mean and the variance of the normalized contour distance function. These features are defined below:

\[
Exp[ND(n)] = \frac{1}{N} \sum_{n=0}^{N-1} ND(n) \tag{6}\]

\[
Var[ND(n)] = \frac{1}{N} \sum_{n=0}^{N-1} (ND(n) - Exp[ND(n)])^2 \tag{7}\]

These features are especially useful in the recognition of "0" and "1", which will be illustrated in part 3.2.2 and 3.2.3.
3.1.2 The normalized length of contour and the normalized area of character

The length of contour of character and the area of character are important features for recognition. In order to reduce the size effect of character, the length of contour and the area of character are normalized by \((h_g + w_d)\) and \(h_g \times w_d\) respectively, which is shown in Fig. 10.

These features are defined below:

\[
\text{average length of contours} = \frac{\text{number of pixels on contours}}{h_g + w_d}
\]

\[
\text{average area} = \frac{\text{number of pixels of character}}{h_g w_d}
\]

3.2 Recognition of handwritten characters using statistical methods

The recognition of handwritten characters is mainly based on statistical analysis, topological properties and contour information. In order to thoroughly make use of contour information, a tree classifier is employed for recognition of handwritten characters, which is shown in Fig. 11. A tree classifier consists of a root node, a number of nonterminal nodes. Tree classifier is a type of multistage classifier. It has advantages over single-stage classifiers. Since a subset of features is selected at each node, and since the features at each node depend on the features extracted at the parent nodes, multistage classifiers use fewer features than single-stage classifiers. This makes a multistage classifiers effective, especially when the number of classes is large and the distribution of the data is multimodal. The recognition process is described below.

3.2.1 Classification of characters with their topological properties

The handwritten characters can be classified into three groups according to their topological properties, which are shown in Figs. 12 to 14. In group one, the handwritten characters only have one contour. It mainly includes 1, 2, 3, 4, 5, 7. In group two, the characters have two contours. It mainly includes 0, 2, 5, 6, 9. In group three, basically there is only one character “8”. Actually, because of complex of writing styles, some unusual characters with three contours are still in this group. These characters are discriminated by the comparison of normalized distance function of outer contour of models and input character.
3.2.2 The further recognition of handwritten characters with one contour

In group one, every character has one contour. The contour information is used for further recognition. The features include: normalized length of contour of characters, normalized area of characters and normalized distance function.

In group one, the simplest character is “1”, which has the shortest relative contour length. In the feature space of normalized length of contour and normalized area of character, which is shown in Fig. 15, we can find that the features of character “1” are in a definitely different area from the features of others. So the decision surface can be calculated easily. If

\[
NA > 1.375(NL - 1.4)
\]

(10)

where NL is normalized length of contours, and NA is normalized area of character, the character will be classified in group “1”. Otherwise, it belongs to “2”, “3”, “4”, “5”, “7”. In group “1”, mainly there is only one character “1”. But because of complexity of writing styles, some characters similar to “1” are still classified into this group. Further classifications are based on the comparison shapes of models and input characters. The rest characters with one contour, which are shown in Fig. 16, are recognized as a shape using normalized distance function. Since there are many variations in handwritten characters, the models of “2”, “3”, “4”, “5” and “7” should be established before the recognition process.

Recognition criteria and establishment of models of one contour characters The recognition of one contour characters is based on the comparison of input character with various kinds of models. The comparison of input character with models is realized by comparing the similarity of their shapes.

we use the similarity function to measure the similarity of two shapes. The similarity function is defined by normalized distance function. In order to overcome the rotation problem, we have to find a reference point, which is invariant to the rotation of a character. In this paper, we use the centroid \((x_c, y_c)\) as the reference point, where

\[
x_c = \frac{1}{n} \sum_{i=1}^{n} x_i
\]

(11)

\[
y_c = \frac{1}{n} \sum_{i=1}^{n} y_i
\]

(12)

where \((x_i, y_i)\) is the i-th pixel on the contour, \(n\) is the number of pixel on the contour.

If the starting point is different, the normalized distance function will be different. In order to make the normalized distance function unique, we have to find the unique starting point. In this paper, we choose the pixel on contour which has the longest distance from centroid as a starting point. The starting point and centroid of “4” are shown in Fig. 16.

For a different character, the number of pixels on contour is always different. In order to derive the similarity function, we have to have the normalized distance function sampled. The criteria of sampling are equal interval and fixed number of samples (In this paper, we choose \(M = 20\)). The normalized distance function after sampling is shown in Fig. 16.

After obtaining the samples of normalized distance function, we can define the similarity function below. The similarity function between input character and \(j\)-th \((j = 1, 2, \ldots, N)\) model is defined as:
The starting point, the centroid of outer contour of "4" and its normalized distance function before and after sampling.

The model of "2" and its normalized distance function.

The model of "3" and its normalized distance function.

The model of "4" and its normalized distance function.

The model of "5" and its normalized distance function.

The model of "7" and its normalized distance function.

\[ \text{Sim}(j) = M - \frac{\sum_{m=1}^{M} \{ND_{\text{input}}(m) - ND^{(j)}_{\text{model}}(m)\}^2}{M} \]  

Here \( ND_{\text{input}}(m) \) is the \( m \)-th sample value of normalized distance function of input character, \( ND^{(j)}_{\text{model}}(m) \) is the \( m \)-th sample value of normalized distance function of \( j \)-th model, \( N \) is the number of models and \( M \) is the number of samples.

The recognition criteria is based on similarity function. The input character will be recognized as model \( J \) if

\[ \text{Sim}(J) = \max \{ \text{Sim}(j), j = 1, 2, \ldots, N \} \]

The models of "2", "3", "4", "5", "7" and their normalized distance function we established are shown in Figs. 17 to 21.

The establishment of models for some special characters using normalized distance function Sometimes a character can be seriously broken and sometimes its topological properties are destroyed. In these cases, a two contour character
becomes a one contour character. So we have to establish some special models for the recognition of these characters. Using the method described before, we can obtain the model of "0", "2", "6", "9" and their normalized distance function, which are shown in Figs. 22 to 25.

Sometimes the topological property of "8" is destroyed. The three contour "8" becomes a two contour character. In this case, we have to establish a special model for the recognition of this kind of characters. Using the similar the method, we can obtain the model of broken "8" and its normalized distance function of outer contour, which are shown in Fig. 26.

### 3.2.3 The recognition of handwritten characters with two contours

In group two, every character has two contours. The information of the first (outer) contour and the second (interior) contour are used for further recognition. The features that we use include the mean and the variance of normalized first (outer) contour distance function, the normalized length of contours of characters, the normalized area of character, the normalized distance of contours, and the position of centroid of contours.

In the feature space of the mean and the variance of normalized contour distance function for 1000
Figure 27: The feature space of the mean and the variance of normalized contour distance function of “0”, “2”, “5”, “6”, “9”.

training data samples, which is shown in Fig. 27. The features of “0” are in an area different from that of the features of other characters. So the decision surface can be found easily. If

\[ \text{Var}[f(n)] < 0.4375 \text{Exp}[f(n)] - 0.6 \]

the characters will be classified into group “0”. Otherwise, it belongs to “2”, “5”, “6”, “9”. In group “0”, mainly there is only one character “0”. But because of complexity of writing styles, some characters similar to “0” are still classified into this group. Further classification are based on the comparison shapes of models of “0” similar characters and input characters.

The above recognition process corresponds to the tree classifier level 2. In level 3, the position of centroid of second (interior) contour are used for recognition of “9”. Because the centroid of second (interior) contour of “9” is always in the upper part of its image, the centroids of second contour of “2”, “5”, “6” are always in the lower part of their images, which is shown in Fig. 28. So the decision principle is established easily. If the centroid of second contour is in the upper part of its image, it will be recognized as “9”. Otherwise, it belongs to “2”, “5”, “6”. The character “5” is recognized at level 4. The character “5” can be recognized by topological relationship between the first contour and the second contour. For character “2” and “6”, their second contour is always inside the first contour. But, for character “5”, its second contour is always outside of first contour. So, the recognition of “5” is very simple. If the second contour is outside of first contour, it will be recognized as “5”. Otherwise, it belongs to “2” and “6”. Their topological relationships are shown in Fig. 29.

At level 5, “2” and “6” can be distinguished by the difference of relative positions of outer and interior contours. From the large amount of experiments, we found that the centroid of interior contour of “2” is always at the left side of the centroid of its outer contour. But the centroid of interior contour of “6” always stays at the right side of the centroid of its outer contour. Our experiments have shown that the discrimination with the relative position of centroids of outer contour and interior contour is very effective. The relative position of centroid of outer and interior contours of characters is shown in Fig. 30. In group “2” and “6”, further recognition are based on the comparison of shapes of models and input characters.

4 Experimental results

In this section, we discuss the experiments we have performed on the handwritten character recognition. The database used in this experiment is NIST database which has more than 20,000 handwritten
characters. Some characters are shown in Fig. 31. We use 1000 normal characters for training, and the other 5073 unseen characters for testing.

In the experiment, we found that there are some confusing characters which are difficult to recognize. These characters are shown in Fig. 32.

The first part of the recognition process in this experiment is shown in Fig. 33. First, the input character is preprocessed. The preprocessing includes: broken character mending, contour smoothing, hole and gap filling, contour extraction etc. Then, the contours of character are followed using TSMA algorithm from outside to inside (or from top to bottom). After that, the input character is classified into three groups according to its topological properties. In order to measure the quality, effectiveness and reliability of recognition of handwritten characters, the following parameters are used in this section.

Recognition rate =

\[
\frac{\text{Number of correctly recognized characters}}{\text{Total number of characters}} \tag{16}
\]

Substitution rate =

\[
\frac{\text{Number of substituted characters}}{\text{Total number of characters}} \tag{17}
\]

Rejection rate =

\[
\frac{\text{Number of rejected characters}}{\text{Total number of characters}} \tag{18}
\]

Reliability =

\[
\frac{\text{Recognition rate}}{\text{Recognition rate} + \text{Substitution rate}} \tag{19}
\]

Further recognition and experimental results are discussed below.

Figure 31: Examples of handwritten characters from the database.
Figure 32: Examples of confusing characters from the database.

Figure 33: The first part of the recognition process.

4.1 The experimental results of recognition of one contour characters

The recognition process for Group 1 is shown in Fig. 34. In this group, every character has only one contour. In order to recognize the character “1”, the length of contour and the area of character are calculated and normalized by the size of the character (height + wide) and the area of the character (height * wide) respectively. If

\[ NA > 1.375(NL - 1.4) \] (20)

where NL is normalized length of contour and NA is the normalized area of character, this character will be recognized as “1”. Otherwise, it will be processed and recognized later.

If the character does not satisfy equation (27), it belongs to “2”, “3”, “4”, “5”, “7”. In order to use the similarity function to recognize these characters, the normalized distance of contour are sampled. Before getting the normalized distance function, the reference point and starting point should be found. We use the centroid of contour as reference point and the point on contour which has the longest distance from centroid as starting point. After that, The similarity (or error) between input character and models are calculated before the final decision is made, and the errors (error(2), error(3), error(4), error(5), error(7)) of the character with respect to the model of “2”, “3”, “4”, “5”, “7” are calculated. Finally, the model with the minimum error are chosen as final recognition result.

4.2 The experimental results of recognition of two and three contour characters

The recognition process of Group 2 is shown in Fig. 35. The features used in this group include: the positions of centroid of contours, the mean and the variance of normalized contour distance function, the normalized length of contours of character, the normalized area of character. For the purpose of recognition and the calculation of distance function, the reference point and the starting point of outer (first) and interior (second) contours should be calculated first. Then, the contour distance function and its mean and variance are calculated. If

\[ Var < 0.4375(Ept - 0.6) \] (21)

this character is classified into group “0”. Otherwise, it belongs to “2”, “5”, “6”, “9”. After that, the centroids of outer and interior contour are calculated. If
Table 3: The final recognition results

<table>
<thead>
<tr>
<th>No. of character</th>
<th>No. of recognized</th>
<th>No. of introduced</th>
<th>Recognition/Restoration rate</th>
<th>Recognition rate</th>
<th>Reliability</th>
</tr>
</thead>
<tbody>
<tr>
<td>total</td>
<td>5073</td>
<td>31</td>
<td>49</td>
<td>98.94</td>
<td>0.21</td>
</tr>
</tbody>
</table>

the centroid of interior contour is at the upper part of its image, it will be recognized as “9”. Otherwise, it belongs to group “2”, “5” and “6”. Then, the topological relationship between the first contour and second contour is used for the recognition of “5”. If the second contour is at the outside of the first contour, it will be recognized as “5”. Otherwise, it belongs to group “2” and group “6”.

The characters “2” and “6” are discriminated according to their relative position of interior and outer centroids. If the interior centroid is on the left of outer centroid, it will be classified into group “2”. Otherwise, it will belong to group “6”.

In Group 3, basically there is only one character “8”. Actually, because of complex of writing styles, some unusual digits with three contours are discriminated by the comparison of the normalized distance function of outer contour of models and input character.

The final recognition results are shown in Table 1. We can see that the recognition of broken handwritten characters using the statistical method is very effective.

5 Conclusion

We have proposed a new statistical recognition method and a new contour following algorithm. Experiments show that the “Two step memory algorithm” (TSMA) is an effective contour following method. It can remove some noises along contours automatically. The recognition process is based on statistical analysis. The features used for recognition are all from various contour information. First, the characters are classified into three groups by their topological properties. In group one, every character has only one contour. The characters in this group are recognized by the features of normalized length of character contour, normalized area of character and distance function. In group two, every character has two contours. The characters in this group are recognized by the relative position of centroids of outer and interior contours, the position of interior centroid, the topological relationship between the first contour and second contour, and the mean and the variance of distance function of outer contour.

Our experiments have shown that this method is very effective. The recognition rate of 5073 handwritten characters which are not used for training is 98.94%.

References


Real-Time Vision Processing for Autonomous Mobile Robots

Gordon Cheng and Alexander Zelinsky
Department of Systems Engineering
Research School of Information Sciences and Engineering
The Australian National University
Canberra, ACT 0200.
Australia
c-mail: Alex.Zelinsky@anu.edu.au

Abstract

In this paper we will discuss three vision-based navigational behaviours for an Autonomous Mobile Robot (AMR). These behaviours are: Collision Avoidance; Obstacle Avoidance; and Goal Seeking. The focus of the discussion will be on the vision processing associated with these behaviours. Experimental results and analysis of the visual processing technique will also be presented.

INTRODUCTION

For many years robotics research has revolved around the idea of having a centralised control architecture that governs the overall processing of the system. Such systems manage a robot’s action in the real world by maintaining a model that represents the operating environment. These systems are often referred to as Planner or Model-based systems[6]. In recent times a new approach to robotics control which is biologically inspired has emerged (referred to as the Behaviour-based[10] approach). This approach grew out of a dissatisfaction with the slow response to real-time events produced by the Planner-based systems. Behaviour-based systems are built from individual competence modules; which involve a strong coupling of a robot’s sensors to its actuators. The end-effect of the collective interaction of the individual modules with the external environment produces a competent control system for the entire robot.

OUR SYSTEM

The configuration of our experimental system as illustrated in Figure 1, comprises of three components: an Off-board vision processing unit; Communication unit; and a Mobile Robot. The vision unit is a Fujitsu MEP vision system. It comes with two processing modules: a video module; and a tracking module. The system was designed to support five tracking modules simultaneously. These modules are connected via a VME-bus backplane to a Motorola MVME-162 Embedded Controller running the VxWorks operating system. The VxWorks operating system handles all program executions. A tracking module can perform up to 100 template matches on each video frame in a 30Hz video stream. Currently we are using one tracking module. The video module provide two selectable input channels, one output channel. All signal are NTSC video format. An NSTC video receiver is connected to one of the input channels on the video module. This receiver is used to accept video signals from the mobile robot.

The communication unit is a SUN-workstation running the Solaris operating system, with a radio modem attached. This manages all network traffic between the vision system and our mobile robot(s). For example, once a image frame is processed, a command from the vision system is send back to the robot, guiding it on its way.

Figure 1: System Configuration

Figure 2: Yamabico Mobile Robot
We are using a Yamabico[9] mobile robot, as shown in Figure 2. It has a multi-processor based architecture, that can house a number of processor modules. All of these modules communicate through the Yamabico-bus, using a Dual-Port-Memory mechanism. The robot has a MC68000-CPU master module, running the Morsa[9] operating system. A T-805 locomotion module, that provides all of the motor feedback and control[12] of the robot. An ultrasonic module is also provided on the robot, this is not used in our experiments. In addition, a radio modem, a small size CCD camera and a video transmitter has been included in our system. The modem is used to communicate with the communication unit. The transmitter and camera provide video input to the vision system.

Information flow

Basic information flow of the system is the robot transmits video signals taken by the on-board CCD camera, the video receiver connected to the vision system receives, which then process them, and send to the SUN workstation. The workstation sends the results back to the robot via the radio modem.

VISUAL BEHAVIOURS

The modular computational structure of a Collision Avoidance behaviour can be simplified into Detection and Avoidance. The detection stage involves determining the availability of free-space in front of the robot for unrestrained motion. If insufficient free space is available the robot surpasses its other behaviours and activates the collision avoidance scheme (eg. "Spin until free space is found"). A situation in which this behaviour can be activated is when the robot is not able to move away from an obstacle, such as a dynamically moving obstacle. Therefore this behaviour acts as a safeguard for the obstacle avoidance behaviour. The Obstacle Avoidance behaviour works by searching for free space within the robot's view of its environment. The robot's basic strategy is to move to where there is free space. The visual processing of the free space search for both of these behaviours will be discussed later. One of the problems with other vision-based avoidance methods (such as optical flow) is their inability to integrate with goal based, navigation and inability to distinguish between a goal and obstacles. The Goal Seeking behaviour exploits the free space searching ability of the vision system. This behaviour allows the robot to focus its attention on searching for a given goal. Once a goal is detected, the robot will perform the associated action for that goal, (eg. Follow or servo).

System Architecture

The system control architecture is illustrated as in Figure 3. Detection modules are being depicted as a diagonal box, actual behaviour modules by a rectangular box and the diamonds are an indication of the associated action. The arrows indicates preconditions to a behaviour, and the line with the small circle indicates inhibition. As shown in Figure 3 the Collision Avoidance has the highest priority over all the other behaviours. The Free-Space-Move needs to be active when the Goal Seeking behaviour is activated.

Navigational behaviours

Navigation can be achieved by using artificial landmarks. By using landmarks no geometric model of the operating environment is needed. The current system can perform artificial landmark based navigation by associating depth information with a given goal. Once a goal is detected, the robot servos towards it. When the robot is 20 cm from the goal it performs an associated action.

Artificial landmarks can be customised to stand out and placed in a location where they can easily be detected, therefore it was ideal for the initial development of our system. In future we would like to develop a system that can acquire natural landmark. The attraction of using natural landmark is that the environment does not need to be modified.

VISUAL PROCESSING

One of the great difficulties in robotics is providing robots with appropriate sensing capabilities, that can supply sufficient information to allow the accomplishment of its task within a reasonable response time. Many sensors such as sonar or infra-red can only provide low-level range information, and cannot provide any kind of goal information. We believe vision can accommodate for this deficiency and provide a robot with the wealth of information its needs to accomplish its tasks more effectively. By using efficient techniques, real-time performance can be achieved. The vision processing methods we are currently using are based on template matching by cross-correlation. Template matching is a simple and effective method for image processing that can allow us to achieve real-time performance.

The free space detection process can be summarised as follows: each image is segmented into a grid of 8 columns and
7 rows, producing 56 individual cells, each cell is 64x64 pixels in size. Each of these cells in the live video is correlated against a stored image of the floor taken at the beginning of each experiment. Due to the distortion of the lens on the CCD camera that we are using, normalisation is performed on the correlation values. After normalising we perform adaptive thresholding of the correlation values, the result of this process determines the amount of free space available to the robot. The goal-detection process utilises the values produced by the normalisation stage of the free-space detection process. An “interest” operator is applied to the normalised values, this operator highlights any features for the goal seeking behaviour to focus its attention on that could be interesting. To verify the correct goal is found a simple reassurance scheme is used. Each of these operations are explained further below.

\[
R(m,n) = \sum_{x} \sum_{y} g(x - m, y - n) - \sum_{x} \sum_{y} f(x,y)
\]

Figure 5: View of the floor taken with the robot's camera

**Template matching**

For each template match, a correlation value is produced ranging from 0 to 65536. This value determines how well a match occurred (the lower the value the better the match). In Figure 4 this is being indicated with a white square, the better the template match the smaller the square. This correlation value is calculated by using Equation (1). The basic assumption of this technique is that the floor the robot is to travel is of constant texture. As with template matching a set of stored templates are needed. To reduce storage space, one grid cell of the floor is stored for each row of templates to be matched. This cell is taken from the center image of a cleared floor. The correlation between the center and the outer edge of the image could vary due to the distortion of the lens. The normalisation stage of the visual processing combats this problem. This can be shown in the following figures, Figure 5 shows a clear floor image taken from the CCD camera mounted on top of the robot. Figure 6 shows a plot of correlation values of this image.

Figure 6: Correlation values (notice lens distortion at the sides)

Figure 7: Normalised and threshold correlation values
Normalisation

Empirically it was determined that a polynomial relation from the correlated values was formed due to the distortion of the camera lens. This can be seen in Figure 6, confidences of each template matches are plotted by the correlated values on each location of the image, the lower the value, the higher the confidence. To overcome this problem, a polynomial curve was fitted to the plot, then for implementation purposes a discrete form as shown in Equation (2) was used to perform the normalisation. At each location of the template match, an associated error value is computed.

\[
E(x, y) = \begin{bmatrix}
    e_{(0,0)} & e_{(0,1)} & \ldots & e_{(0,n-1)} \\
    \vdots & \ddots & \vdots & \vdots \\
    e_{(m-1,0)} & e_{(m-1,1)} & \ldots & e_{(m-1,n-1)}
\end{bmatrix}
\]

\[
R(x, y) = \begin{bmatrix}
    -1 & -1 & -1 \\
    -1 & 8 & -1 \\
    -1 & -1 & -1
\end{bmatrix}
\] (4)

Free-space

Free-space is determined by a threshold value, this is calculated by using the new correlated values produced from the normalisation stage. This threshold value is determined using the variances between all the normalised correlated values. The is given by Equation (3). This threshold is then applied to the normalised correlated values, the results can be seen in Figure 7.

\[
\sigma = \frac{\sum f(x, y) - \bar{f}}{n - 1}
\] (3)

Interest-Operator

For goal detection, the original system described in [7] searches the complete images for a given goal, it was noticeably inefficient. An interest operator was introduced to increase efficiency of the goal seeking behaviour by reducing the search space. This interest-operator can also be used as a feature detector that doesn’t use an explicit model, which leads to landmark acquisitions.

The goal detection processing is as follow. An image from the video camera is shown in Figure 8. This image is then processed through the correlation step as described above. This produces correlation values as shown in Figure 9. Normalisation is then performed on the values, as shown in Figure 10. The final step is applying the Interest-operator, the effect which can be seen in Figure 11. The Interest-operator produces a single peak that can be used to easily identify a feature of interest in the robot’s view.

The Interest-operator as shown in Equation (4) is used. The effect of this operator produces a second derivative from the correlated values.
Recognition/Reassurance

This is the final stage of the goal detection process, this process reassures, if needed, that the goal detector is looking at an object that it is seeking. Before the execution of the experiment five templates of a goal are stored, all five templates are taken from various orientations of the same goal.

Noise filtering

Filtering was also performed to overcome the problems caused by noisy transmission of the video signals to the vision system. The filter performs sanity checking on all the commands that are sent to the robot, this checking is done by a voting scheme. This ensures the robot does not get into a state such start-stop motion in between consecutive video frames.

CONCLUSION

Template matching is an efficient method for achieving real-time vision processing.

From our experimentation of combining the visual behaviours in various configurations interesting overall behaviours have been produced. For example by combining the Collision and Obstacle Avoidance behaviours we obtain wandering and survival, this can be shown in Figure 13. The Wandering behaviour is fundamental to many exploration experiments, such as map building, and landmark acquisition. By combining Goal Seeking with the overall Survival behaviour we can produce experiments in Landmark Navigation, as shown in Figure 12. All these behaviours exhibit real-time performance which is robust and reliable in both static and dynamic environments.

In normal operation the current system can perform its task at a robot velocity of up to 500 mm/sec.

REFERENCES


Interaction Between Topological and Interval Relationships in Two-Dimensional Space

Mohammad Nabil
School of Computer Science and Engineering
The University of New South Wales
Sydney 2052, AUSTRALIA
nabil@cse.unsw.edu.au

Abstract

Both topological and directional relationships are important properties that can be used to reason about spatial objects as well as to express constraints for spatial objects retrieval. Integration of these relationships certainly increase the expressive power of qualitative spatial relationships. This paper investigate interaction between topological and interval relationships to build a simple and elegant algorithm for constructing 2D Projection Interval Relationship (2D-PIR): an integration of topological and directional relationships. The paper also discuss about alternative definition of directional relationship derived from 2D-PIR representation.

Keywords Spatial Relationships, Spatial Reasoning, 2D-PIR

1 Introduction

Topological and directional relationships have attracted many researchers in different fields including spatial information systems and multimedia information systems. These relationships are central to spatial reasoning. They are also very useful for expressing constraints in image retrieval based on spatial objects.

Both topological and directional relationships have been explored in depth by many researchers such as in [6, 2] for directional relationships and in [4, 3, 9] for topological relationships. Although the integration of topological and directional relationship offers great expressive power for describing spatial relationships, there are very few attempts focussing on this integration. For example, with only directional relationships, we can only say that Queensland is to the north of New South Wales, and with only topological relationships, we can only say that Queensland and New South Wales have a common border, but until we can deal with both kinds of relationship we cannot reason about Queensland being a northern neighbour of New South Wales.

In [5] we attempted to integrate topological and directional relationships into a unified representation called 2D Projection Interval Relationships (2D-PIR). Basically, 2D-PIR integrates the work of Egenhofer [4], Allen [1] and Chang [2] to represent spatial relationships in a 2D space. However, we have not explored the interaction between Egenhofer’s topological relationships and Allen’s interval relationships. In this paper we concentrate on this interaction since we found that Egenhofer’s topological relationships, to some extent, interact with Allen’s interval relationships in 2D space. There are two interesting aspects of this interaction. First, we can represent 2D-PIR more compactly since all redundancy can be removed. Second, a simpler algorithm for computing topological relationships and 2D-PIR on regions can be developed.

The remainder of this paper is structured as follows. Section 2 presents related work. In section 3 we review our existing representation of spatial relationships (2D-PIR) and then discuss the interaction between topological and interval relationships components of 2D-PIR. In Section 4 we develop algorithm for computing our 2D-PIR relationship based on interaction between topological and interval relationships. Section 5 discusses how to derive directional relationships from existing 2D-PIR relationships. Finally we present our conclusions in section 6.

2 Related Work

Egenhofer [4] uses the concepts of boundary and interior of pointsets to derive topological relationships between two-dimensional regions (pointsets) embedded on a two-dimensional plane ($\mathbb{R}^2$). If $A$ and $B$ are two regions, $A^0$ and $\partial A$ denote the interior and boundary of $A$ respectively, and all of the possible topological relationships between $A$ and $B$ can be derived from the possible combinations of intersection between their boundaries and interiors; that is $\partial A \cap \partial B(1), A^0 \cap B^0(2), \partial A \cap B^0(3)$, and $A^0 \cap \partial B(4)$. Each of these intersections is either empty ($0$) or non-empty ($1$). This results in 16 possible combinations as shown in Table 1.

There are nine valid topological relationships between two regions. But $r_5$ and $r_9$, $r_{13}$ and $r_{14}$
<table>
<thead>
<tr>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>Relationships' Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>$r_0$</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>disjoint</td>
</tr>
<tr>
<td>$r_1$</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>$r_2$</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>$r_3$</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>$r_4$</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>$r_5$</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>$r_6$</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>$r_7$</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>$r_8$</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>$r_9$</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>$r_{10}$</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>$r_{11}$</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>$r_{12}$</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>$r_{13}$</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>$r_{14}$</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>$r_{15}$</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>

*) indicates there is no topological relationship (proved in [4]).

Table 1: Topological relationships created from four intersections

are inverses of each other. There are two overlaps relationships ($r_7$ and $r_{15}$) since disconnected boundaries are allowed. If boundaries are assumed to be connected, then $r_7$ is excluded. Hence, there are eight mutually exclusive topological relationships: disjoint, contains, insides, overlaps, meets, equal, covers, and coveredBy.

Allen [1] proposed an interval-based temporal representation and introduced a method to derive relationships between intervals. Intervals can be represented by their endpoints assuming that every interval consists of a fully ordered set of points of time. Thus an interval is represented as an ordered pair of points with the first point less than the second, and so on. Based on this representation, thirteen relationships (seven relationships have inverses, one relationship has no inverse) are derived as shown in Table 2.

<table>
<thead>
<tr>
<th>Relationship</th>
<th>Symbol</th>
<th>Symbol for Inverse</th>
<th>Pictorial Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>$X$ before $Y$</td>
<td>$&lt;$</td>
<td>$&gt;$</td>
<td>XXXXX YYYY</td>
</tr>
<tr>
<td>$X$ equal $Y$</td>
<td>$=$</td>
<td>$=$</td>
<td>XXXX YYYY</td>
</tr>
<tr>
<td>$X$ meets $Y$</td>
<td>$m$</td>
<td>$m$</td>
<td>XXXXXXXXXXX</td>
</tr>
<tr>
<td>$X$ overlaps $Y$</td>
<td>$o$</td>
<td>$o$</td>
<td>XXXXXX YYYY</td>
</tr>
<tr>
<td>$X$ during $Y$</td>
<td>$a$</td>
<td>$a$</td>
<td>XXXXXXX YYYY</td>
</tr>
<tr>
<td>$X$ starts $Y$</td>
<td>$s$</td>
<td>$s$</td>
<td>XXXXX YYYY</td>
</tr>
<tr>
<td>$X$ finishes $Y$</td>
<td>$f$</td>
<td>$f$</td>
<td>XXXXXXX YYYY</td>
</tr>
</tbody>
</table>

Table 2: Allen's interval relationships

Chang et al. [2] proposed the 2D-String to represent spatial relationships in 2D-space. The 2D-string uses a symbolic projection of a picture along the $x$-axis and the $y$-axis. The 2D-string representation for an image is generated by first placing a grid over the image; in this grid, some slots contain objects or parts of objects as depicted in Figure 1.

<table>
<thead>
<tr>
<th>DE</th>
<th>B</th>
<th>C</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>A</td>
<td></td>
</tr>
</tbody>
</table>

Figure 1: A symbolic picture

A 2D-string is a pair made up of the "symbolic projections" of a picture along the $x$-axis and $y$-axis. Construction of a 2D-string for a picture commences by laying a grid over the picture; the idea is that objects are partitioned over the cells of the grid (as in Figure 1). To generate a 2D-string from a picture, we scan along the $x$-axis, looking at the projections of objects on this axis, and describing the relationships between the objects using the operators "$<", "=", "<" and ">" "$=$" represents the relationship "at the same projected location as", "$<$" represents the relationship "to the left of" and "$>$" represents "in the same set as" relation (note that "$=$" implies "$="). For example, in Figure 1, the $x$-axis relationships are: $A = D : E < A = B < C$. We then repeat this scan along the $y$-axis, to generate another string of relationships between objects (or, more precisely, their projections on the $y$-axis). In the example, this corresponds to: $A = A < B = C < D : E$. The 2D-string of the picture is $(A = D : E < A = B < C, A = A < B = C < D : E)$

3 An Integration of Topological and Directional Relationships

3.1 2D Projection Interval Relationships

A 2D Projection Interval Relationship (2D-PIR) is a symbolic representation of directional as well as topological relationships among spatial objects. It adapts three existing representation formalisms (Allen’s temporal intervals, Egenhofer’s topological relationships and 2D-strings) and combines them in a novel way to produce the unified representation. Note that in this paper we only deal with 2-dimensional spatial objects; higher dimensional spatial objects appear to be a straightforward extension of the ideas presented here. The basic idea is that each spatial object is projected along the $x$ and $y$ axes forming an $x$-interval and a $y$-interval for the object. Using the intervals and Allen’s 13 interval relationships, spatial relationships are established.

2D-PIR also uses the projection concept from the 2D-string representation. It differs from the 2D-string representation in using Allen’s interval relationships over the projections of the entire objects along the $x$ and $y$ axes, rather than 

<.
There is thus no need to partition the image initially. A consequence of this approach, however, is that we must use a graph, rather than a string, to represent the relationships among the objects in a space since some of Allen’s relationships are not transitive.

**Definition 1** A 2D-PIR between two spatial objects A and B is a triple (δ, χ, ψ) where δ represent topological relationships, χ and ψ represent relationships between object projected along the x-axis, and ψ respectively.

In Figure 2(a) the 2D-PIR relationship between A and B is (to, di, oi). The 2D-PIR relationship between B and A is the inverse of (to, di, oi) which is (to, d, o). In Figure 2(b) the 2D-PIR relationship between C and D is (ct, di, di).

![Figure 2: A symbolic picture](image)

### 3.2 Interaction among Components in 2D-PIR Relationships

The initial idea of 2D-PIR relationship is represented by (χ, ψ). This representation has a serious drawback since it works only for rectangular objects. It is true that we can represent spatial objects using their bounding rectangle, however, if bounding rectangles of spatial objects overlap then the topological relationships derived cannot be guaranteed correct. To overcome this problem we introduce Egenhofer’s topological relationships into 2D-PIR representation. Adding topological relationships (δ) makes 2D-PIR robust. Note that all topological relationships defined by Egenhofer may not be needed since some of them can be implied.

Based on the interaction between topological and 2D interval relationships, we can extract five basic topological relationships in term of 2D-PIR namely disjoint (dt), touch (to), inclusion (ic), equal (eq) and overlap. Other topological relationships can be implied from ic relationship and interval relationships (χ and ψ). The disjoint relationship (dt) is basic since this relationships cannot be implied from any topological relationship combine with interval relationship χ or ψ. Disjoint may exist for any combination of χ and ψ. Yet, the interval relationships before and after imply disjoint. The later property is quite useful in deciding whether two regions (polygons) are disjoint. If the projection of polygons (on x-axis or y-axis) has relationship before (<) or after (>) then the two polygons are disjoint. However we can not conclude that the relationship between polygons is not disjoint if their projection (either on x-axis or y-axis) is not before or after as stated above. Figure 3 is an example where two polygons are disjoint but their χ and ψ ∉ {<, >}.

![Figure 3: A and B are disjoint](image)

Similarly, touch and meet are basic topological relationships in 2D-PIR. The touch relationship may exist for any combination of χ and ψ. The interval relationship meet (m) and meetInverse (mi) does not imply the touch relationships and vice versa.

The new relationship is inclusion (ic). Two spatial object is inclusion if the interior of one object is a subset of the other. We can deduce other topological relationships (inside, contain, covers, coveredBy) using is and some combination of χ and ψ.

The next basic relationship is equal (eq). Actually eq is a special case of ic. But, this relationship cannot be deduced from the ic and any combination of χ and ψ. Therefore we also consider equal is a basic topological relationship in 2D-PIR.

The last basic relationship is overlap. The reason to include overlap as a basic topological relationship in 2D-PIR is that this relationship cannot always be implied from the combination of ic, χ and ψ as depicted in Figure 4. In this figure, χ and ψ between A and B in (a) and (b) is the same. The topological relationship between A and B in (b) is overlap which clearly cannot be deduced from ic combined with χ and ψ. In conclusion, the domain of δ in 2D-PIR: (δ, χ, ψ) is {dt, to, ic, eq, ov}.

![Figure 4: (a) A inside B (b) A overlap B](image)

Our basic topological relationships above give a strong foundation for Clementini [3] to generalise
topological relationships into five general relationships.

As we mentioned above, other topological relationships can be implied from the basic topological relationships combined with interval relationships ($\chi$ and $\psi$). The following rules deduce the rest of topological relationships (covers, coveredBy, inside, and contain) by combining the relationship is with $\chi$ and $\psi$.

1. Rule #1: if $\delta = ic \land \chi \in \{s,f\} \land \psi \in \{s,d,e\}$ then the topological relationship is coveredBy.
2. Rule #2: if $\delta = ic \land \chi \in \{s,f,d,e\} \land \psi \in \{s,f\}$ then the topological relationships is coveredBy.
3. Rule #3: if $\delta = ic \land \chi \in \{s,f,i\} \land \psi \in \{s,i,d,i,=\}$ then the topological relationship is covers.
4. Rule #4: if $\delta = ic \land \chi \in \{s,f,i,d,i,=\} \land \psi \in \{s,f,i\}$ then the topological relationship is covers.
5. Rule #5: if $\delta = ic \land \chi = d \land \psi = d$ then the topological relationship is inside.
6. Rule #6: if $\delta = ic \land \chi = d \land \psi = d$ then the topological relationship is contain.

4 Algorithm for Computing 2D-PIR Relationship between Polygons

Polygons are modeled as an ordered set of points where the first point is the same as the last point. To compute a 2D-PIR relationship between two polygons there are two steps. The first step is to compute $\chi$ and $\psi$ and the second step is to compute $\delta$. It is important to compute $\chi$ and $\psi$ first since in some cases we can avoid intensive computation of $\delta$. For example if $\chi$ or $\psi \in \{<,>\}$ then immediately we can decide that $\delta = dt$.

Let $P_1$ and $P_2$ be two polygons represented as array of points. The following algorithm computes components of the 2D-PIR. To construct a 2D-PIR relationship between polygons, we need only to integrate the results of the first step and the second step.

ALGORITHM: Compute 2D-PIR relationships between two polygons

INPUT: Two polygons ($P_1$ and $P_2$)

OUTPUT: 2D-PIR relationship between $P_1$ and $P_2$

/* The first step: Compute interval relationship */
begin /* first step */
Compute minimum and maximum value both for $P_1$ and $P_2$
Let $s_1, f_1, s_2, f_2$ be the minimum and maximum values of intervals of $P_1$ and $P_2$. (Note that interval can be interval projection of a polygon on x-axis or y-axis) and $R$ be an interval relationship.
if $(f_1 < s_2)$ then $R = before$
else if $(s_1 > f_2)$ then $R = after$
else if $(f_1 == s_2)$ then $R = meet$
else if $(s_1 == f_2)$ then $R = meetInverse$
else if $(s_1 < s_2)$
if $(f_1 < f_2)$ then $R = overlap$
else if $(f_1 == f_2)$ then $R = finishInverse$
else then $R = duringInverse$
else if $(s_1 == s_2)$
if $(f_1 < f_2)$ then $R = start$
else if $(f_1 == f_2)$ then $R = equal$
else then $R = startInverse$
else if $(s_1 > s_2)$
if $(f_1 < f_2)$ then $R = during$
else if $(f_1 == f_2)$ then $R = finish$
else then $R = overlapInverse$
return then $R$
end /* first step */

/* The second step: Compute topological relationship $\delta$ of $P_1$ and $P_2$ */
begin /* second step */
compute $\chi$ and $\psi$ using the first step algorithm
if $\chi \in \{<,>,>,\}$ or $\psi \in \{<,>,>,\}$ then $\delta = dt$
else
compute polygon intersection between $P_1$ and $P_2$
if the result is disjoint then $\delta = disjoint$
else if the result is overlap then $\delta = overlap$
else if result is share boundaries but their interiors are disjoint then $\delta = touch$
else $(P_1^0 \subset P_2^0$ or $P_2^0 \subset P_1^0$)
if their boundaries are not the same then $\delta = inclusion$
else $\delta = equal$
return $\delta$
end /* second step */

The worst case complexity of the first step (compute interval relationships) is $O(n)$ where $n$ is the number of points representing polygons. This is mainly for computing the minimum and maximum value of intervals.

The worst case complexity of the second step (compute basic topological relationships) is $O(mn)$ where $m$ and $n$ are number of points representing polygons $P_1$ and $P_2$ respectively.

We implemented the algorithm above using the Java Language. The following figures are the results of some experiments with the algorithm. Note that in the program we encode 2D-PIR relationships using hexadecimal integers. Recall, the symbol of 2D-PIR is $(\delta, \chi, \psi)$ where $\delta$ is a topological relationship (0-disjoint, 1 touch, 2 inclusion, 3-equal, 4-overlap), $\chi$ and $\psi$ are interval projection relationships on x-axis and y-axis respectively (1-
before, 2-meet, 3-overlap, 4-finishInverse, 5-start, 6-duringInverse, 7-during, 8-startInverse, 9-finish, a-overlapInverse, b-meetInverse, c-after)

In Figure 5, Figure 8 and Figure 9, the algorithm detects that two objects (polygons) are disjoint ($\delta = 0$). In Figure 5 and Figure 6 the 2D-PIR relationships between the small and the big objects is (0,3.7) and (4,3.7) respectively. In terms of interval projection on the x-axis and y-axis both pictures are same which is 3 (overlap) and 7 (during). But their topological relationships are different: in Figure 5 both objects are disjoint but in Figure 6 both objects overlap.

Consider Figure 7 and Figure 8. The 2D-PIR relationships between the small and big objects are (2,6.6) and (0,6.6) respectively. In Figure 7 the small object is inside the big one (2,6.6 is inside - rule #5) but in Figure 8 the small object is disjoint with the big one (it seems small object surrounded by the big one). Finally in Figure 10 the algorithm correctly detects the touch relationship between two objects (2D-PIR : (1,6,2)).

We tested our algorithm against more than one hundred cases. As we expected, the algorithm is robust.

![Figure 6: Experiment # 2](image)

![Figure 7: Experiment # 3](image)

5 Rules for Directional Relationships

Penquet and Ci-Xiang [6] have developed a model for directional relationships between polygons by incorporating size, shape and distance. They also provide an algorithm to determine directional relationships between polygons in 2D space. In [8], fuzzy concepts were introduced into directional relationships, to account for the fact that such relationships are not necessarily exclusive. For example, if an object is above and to the left of another object, it may be perceived as being above, leftOf, or some degree of both of these.

Sistla [7] defines directional relationships using x and y coordinates. In this scheme, for example, A is leftOf B in picture p if the x-coordinate of every point of A in p is less than the x-coordinate of every point of B in p. Similarly A is above B in picture p if the y-coordinate of every point of A in p is greater than the y-coordinate of every point of B in p.

Our 2D-PIR relationship provides a novel concept to define directional relationships not only between two objects, but also directional relationships between parts of objects. If we look at the map of Europe, generally one may say that Portugal is west of Spain. But one may argue that Portugal is not only west of Spain but also is south of Spain or even north of Spain since there exists part of Spain is north of Portugal as well as south of Portugal. This issue can not be resolved using traditional directional models such as mentioned above. 2D-PIR solves this problem easily.
Figure 8: Experiment # 4

Figure 9: Experiment # 5

For the purpose of clarity we use directional relationship rightOf-leftOf (above-below) rather than west-east (north-south) since the reference object for rightOf-leftOf relationship is the one who see the objects. But certainly rules that we develop below can also be applied to west-east relationship. We have eight directional relationships between objects in 2D space namely rightOf, leftOf, above, below, aboveLeft, aboveRight, belowLeft and belowRight. We will describe in detail these relationship using rules. Before we discuss these rules we introduce the notion of total and partial directional relationships.

**Definition 2** Two objects are in a total directional relationship if the directional relationships between them can be explained by one directional relationship. Two objects are in a partial directional relationship if the directional relationship between them cannot be explained by one directional relationship.

Figure 10: Experiment # 6

Figure 11: Total and partial directional relationship

The following are rules for deriving directional relationships from 2D-PIR.

1. Rule #7: if \( \delta \in \{dt, to\} \land \chi \in \{<, m\} \land \psi \in \{<, m, o, d, f, \ldots, fi, di, si, oi, mi, >\} \) then the directional relationship is leftOf.

2. Rule #8: if \( \delta \in \{dt, to\} \land \chi \in \{>, mi\} \land \psi \in \{<, m, o, d, f, \ldots, fi, di, si, oi, mi, >\} \) then the directional relationship is rightOf.

3. Rule #9: if \( \delta \in \{dt, to\} \land \chi \in \{<, m, o, d, f, \ldots, fi, di, si, oi, mi, >\} \land \psi \in \{<, m\} \) then the directional relationship is below.

4. Rule #10: if \( \delta \in \{dt, to\} \land \chi \in \{<, m, o, d, f, \ldots, fi, di, si, oi, mi, >\} \land \psi \in \{>, mi\} \) then the directional relationship is aboveRight.
6. Rule #12: if $\delta \in \{dt, to\}$ and $\chi \in \{<, m\}$ and $\psi \in \{>, mi\}$ then the directional relationship is belowRight.

7. Rule #13: if $\delta \in \{dt, to\}$ and $\chi \in \{<, m\}$ and $\psi \in \{<, m\}$ then the directional relationship is aboveLeft.

8. Rule #14: if $\delta \in \{dt, to\}$ and $\chi \in \{>, mi\}$ and $\psi \in \{>, mi\}$ then the directional relationship is belowLeft.

The above rules define total directional relationships. Partial directional relationships are omitted if $\delta \in \{dt, to\}$ and both $\chi \land \psi \in \{o, i, f, s, fi, di, ai, si\}$. For instance the directional relationship between A and B in Figure 11(d) is (dt, o, i) which is a partial directional relationship. We can derive a new relationship from partial directional relationships. For example the relationship (dt, d, d) such as depicted in Figure 8 is a partial directional relationship which describes how an object encompasses another object.

6 Conclusion

In this paper we investigated the interaction between Egenhofer’s topological relationship and Allen’s interval relationships in two-dimensional space. We found that there are some interactions between Egenhofer’s and Allen’s relationships. With this interaction we can generalise Egenhofer’s topological relationships into five basic relationships in terms of our 2D-PIR representation. The interaction also leads to a simple and elegant algorithm for computing 2D-PIR relationships between spatial objects in 2D-space (polygons).

The 2D-PIR relationship representation, which integrates topological and directional relationships, is certainly more powerful than traditional representations of spatial relationships. Using the 2D-PIR representation we can derive not only directional relationships between whole spatial object but also directional relationships between parts of objects. In addition, it is possible to derive new spatial relationships between objects in a clear and concise way.

In the future, we plan to investigate the application of 2D-PIR relationships in the development of a spatial query language.

References


Dynamics Analyses of Recurrent Systems and their Application to Pattern Recognition

Alan M. N. Fu and Hong Yan
Department of Electrical Engineering
University of Sydney, NSW 2006, Australia
phone: +61 2 9351-4824
fax: +61 2 9351-3847
e-mail: alanf@ee.usyd.edu.au

Abstract

Two kinds of recurrent systems, i.e., the Hopfield-Amari networks (HAN) and probabilistic relaxation schemes (PRS) are considered in this paper. We will first show that the theoretical analysis of HAN in our previous work can be used to predict the stable states of the system. Then a new recurrent system based on a PRS is developed for pattern recognition. The PRS is established in terms of both the probability space partition and the complete probabilistic formula. It overcomes the flaws that arise in some existing relaxation schemes. The essence of the proposed system in solving a pattern recognition problem is the following: (1) to construct a probability space in the features space of the related pattern recognition problem; (2) to generate a partition of the probability space. The probabilistic distribution of the partition of the probability space will serve as an initial input of the system, and the system becomes an effective classifier of patterns after simple training. The experiments performed on shapes recognition to verify the effectiveness and reliability are also given. The results show that the proposed system is very useful in solving various pattern recognition problems.

1 Introduction

Recurrent systems, i.e., synchronous feed back systems, have a simple structure, but have many interesting dynamical properties and equilibrium. This kind of Recurrent system includes Hopfield-Amari networks (HAN) and probabilistic relaxation schemes (PRS) which have been considered through theoretical analyses, application and experiments since 1970’s by many researchers, such as Amari et al. [1], Amari and Maginu [2], Zagreblov and Chvyrov [3], Patrick and Zagreblov [4] [5], Nishimori and Ozaki [6], Fu [7], Fu and Yan [8], Rosenfeld et al. [9], Zucker et al. [10], Polo and Rosenfeld [11], Polo [12], Chen and Luh [13]. In this paper, we first show that the theoretical results in Fu [7] are not only effective in predicting the dynamics of HAN at several earlier steps in a recalling process, but also useful in predicting the stable states of the system with a small or large pattern ratio. Then a new PRS based on the idea of a probability space partition and the complete formula is presented, which overcomes the flaws that arise in some existing relaxation schemes. Finally, experiments on shape recognition are performed to verify the effectiveness and reliability of the proposed PRS.

2 Hopfield-Amari Networks

A Hopfield-Amari Network (HAN), a synchronous recurrent neural network (system) with inter-connection weights determined by Hebbian rule [14], is considered to be a 4 variables (mean \(\mu(t)\) of the main overlap, mean \(\mu_i(t)\) of the main overlap and variance \(\sigma_A(t)\) of the main overlap and variance \(\sigma_N(t)\) of noises) stochastic system. It has been shown that the dynamics of HANs can be described by the following two theorems [7].

**Theorem A**

Suppose the following three conditions are satisfied:

(i) the components \(X_i(t), i = 1, 2, \ldots, n\), of the state of the system are exchangeable random variables;

(ii) the noise terms \(N_i(t), i = 1, 2, \ldots, n\), are independent exchangeable identical normal random variables with expectation \(\mu_N(t)\) and standard deviation \(\sigma_N(t)\);

(iii) \(P(A(t) = a(t)) = 1\);
then the governing equations for the macroscopic state of the neural network are given by

\[ a(t+1) = F(a(t)/\sigma_N(t)), \]
\[ \mu_N(t+1) = 2r\sigma_N^2(t)a(t)p(-\eta(a(t),t)), \]
\[ \sigma_N^2(t+1) = r + 4p^2(\bar{a}(t)) + 4rp(\bar{a}(t))\bar{a}(t)a(t+1), \]

where

\[ a(t) = E[A(t)], \text{ the expectation of } A(t), \]
\[ r = m/n, \text{ the pattern ratio}, \]
\[ F(x) = \Phi(x) - \Phi(-x), \text{ } -\infty < x < \infty, \]
\[ \Phi(y) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{y} p(x) \, dx, \]
\[ \eta(x,t) = \frac{x + \mu_N(t)}{\sigma_N(t)}, \]
\[ \bar{a}(t) = \frac{a(t)}{\sigma_N(t)}, \]
\[ p(x) = \frac{1}{\sqrt{2\pi}} \exp(-\frac{1}{2}x^2). \]

**Theorem B**

If the assumptions (i) and (ii) in Theorem A together with the following two conditions are satisfied:

(iii) the main overlap \( A(t) \) is normally distributed with expectation \( a(t) \) and standard deviation \( \sigma_A(t) \);

(iv) the main overlap \( A(t) \) and noise terms \( N_i(t), i = 1, 2, \ldots, n, \) are independent random variables;

then the recurrent equations for the macroscopic state of the neural network are given by

\[ a(t+1) = \int_{-1}^{1} p_A(x,t)F(\eta(x,t))dx, \]
\[ \sigma_A^2(t+1) = \int_{-1}^{1} p_A(x,t)F^2(\eta(x,t))dx - a^2(t+1), \]
\[ \mu_N(t+1) = 2r\sigma_N^2(t)a(t)p(-\eta(a(t),t)), \]
\[ \sigma_N^2(t+1) = r + 4p^2(\bar{a}(t)) + 4rp(\bar{a}(t))\bar{a}(t)a(t+1), \]

where

\[ p_A(x,t) = \frac{1}{\sqrt{2\pi}\sigma_A(t)} \exp(-2^{-1}(x - a(t))^2\sigma_A^{-2}(t)). \]

Theorem B reduces to Theorem A if the density function \( p_A(x,t) \) is a \( \delta \)-function, and if \( \mu_N(t) \equiv 0 \) then Theorem A reduces to Amari’s theorem 2 [2]. In this paper, we will show that the proposed theorems are not only effective in analyzing the dynamics of the HAN at several earlier steps in a recasting process, but also successful in predicting its stable states, which is confirmed by the comparisons between the theoretical analyses and experiments shown in Figs. 1 and

2. Thus, our results are an improvement on Amari’s theory [2], the HAN has been successfully applied to pattern recognition [16-18].

![Figure 1: Comparison between theoretical analyses and experiments of HAN with \( r=0.08 \) and \( a(0)=0.7 \).](image1)

![Figure 2: Comparison between theoretical analyses and experiments of HAN with \( r=0.3 \) and \( a(0)=0.7 \).](image2)

## 3 Probabilistic Relaxation Schemes

In terms of the complete probability formula and the partition of a probability space a new PRS has been developed [15], in which the measure of inconsistency between the certainty and compatibility of the system is completely determined by the probability theory, and therefore it overcomes the flaws (such as, the independent labeling problem, the non-independent labeling problem [12] [14]) that arise in some existing Relaxation Schemes, such as Rosenfeld [9], Zucker et al. [10], Peleg and Rosenfeld [11], Chen and Luh [13]. Although Peleg’s method [12] does not have these flaws, it is very complex in calculation and some approximate procedure needed. In this paper, we will show that the PRS presented in [14] not only overcomes the flaws but is also effective in solving pattern recognition problems.

The PRS is described as follows. Let \( g_l, l = 1, \ldots, n \) denote the \( n \) objects that we want to classify into \( m \) different classes. Suppose that the fea-
tures space of each object is known. Let \( G_i \) denote the probability space of \( g_i \) resulting from its features space, and \( \{G_i(1), \ldots, G_i(m)\} \) is a partition of \( G_i \) and \( p_i^{(0)}(k) \) is the probability of \( G_i(k) \). Thus, the object \( g_i \) corresponds to the probability vector \( p_i^{(0)} = (p_i^{(0)}(1), p_i^{(0)}(m))^T \), which is the initial certainty measure vector (ICMV) of object \( g_i \). The formulation of our PRS is described by the following recurrent equation:

\[
p_i^{(k+1)}(\lambda) = p_i^{(k)}(\lambda) + \frac{p_i^{(k)}(\lambda)(p_i^{(k)}(\lambda) - s_i^{(k)}(\lambda))}{R_i^{(k)}}
\]

where

\[
s_i^{(k)}(\lambda) = \sum_{j=1}^{n} \sum_{\lambda'=1}^{m} c_{ij}(\lambda, \lambda') p_i^{(k)}(\lambda'),
\]

\( c_{ij}(u, v), j, l = 1, \ldots, n \) are compatibility coefficients, which satisfy the condition: \( \sum_{u=1}^{n} c_{ij}(u, v) = 1 \). \( d_{ij} \) is a positive coefficient which satisfies: \( \sum_{j=1}^{n} d_{ij} = 1 \).

The principle of the PRS and its function are illustrated in Fig. 3.

![Figure 3: The Principle of PRS and its Function.](image)

In order to pictorialize the properties of PRS, we consider the PRS with \( m = 3 \) and call it a 3-dimensional system. Since \( \sum_{z=1}^{3} p_i^{(k)}(z) = 1 \), we can regard the vector \( p_i^{(k)} \) as a point in the triangle plane \( G^3 = U_1 U_2 U_3 \) in 3-dimensional space (Fig. 4), where \( U_1, U_2 \) and \( U_3 \) are 3 unit vectors, i.e., \( U_1 = (1, 0, 0) \), \( U_2 = (0, 1, 0) \) and \( U_3 = (0, 0, 1) \). Each point on \( G^3 \) corresponds to a labeled object whose initial certainty measures are given by the components of the coordinate of the point. So there are an infinite number of labeled objects. In our experiment, there are 210 labeled objects which are distributed uniformly over a grid of \( G^3 \). Their initial certainty measure vectors, \( p_i^{(0)} \), \( i = 1, \ldots, 210 \), will be processed by the relaxation rule in Eq. (2). The relaxation process iteratively modifies the certainty measure vector \( p_i^{(k)} \) and generates the final certainty measure vector \( p_i^{(\ast)} \). Based on the nature of \( p_i^{(\ast)} \), the grid point on \( G^3 \) is marked with one of the five pre-defined symbols, "1", "2", and "3" for \( U_1, U_2 \) and \( U_3 \), respectively, and "\( \ast \)" for the vector \( p_i^{(k)} \) which does not change during iteration, and "\( \ast \)" which means that \( p_i^{(k)} \) is not a unit vector of the 3-dimensional space, but a fixed point on \( G^3 \). A triangle diagram with the above symbols is obtained after a finite number of iterations of PRS for a given compatibility matrix \( C \). Figs. 5-7 show the comparisons between Fu-Yan’s and Rosenfeld’s PRSs. It is clear that our scheme has overcome the flaws that arise in Rosenfeld’s model (including the models in [10] and [13]).

![Figure 4: All possible labeled objects in 3-dimensional space.](image)

![Figure 5: Comparison between Fu-Yan’s and Rosenfeld’s PRS.](image)
Table 2: The FCMVs of models (Fig. 7) with $C^*$

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
</tr>
</thead>
<tbody>
<tr>
<td>$p_1^*$</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>1.00</td>
</tr>
<tr>
<td>$p_2^*$</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>1.00</td>
</tr>
<tr>
<td>$p_3^*$</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>1.00</td>
</tr>
<tr>
<td>$p_4^*$</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>$p_5^*$</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
</tr>
<tr>
<td>$p_6^*$</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>$p_7^*$</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>$p_8^*$</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>$p_9^*$</td>
<td>1.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>$p_{10}^*$</td>
<td>1.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
</tbody>
</table>

Table 2 shows the final certainty measure vectors (FCMV) of model shapes by PRS with uniform $C^*$. We can see that all the FCMVs of models converge to the correct unit vector in this case.

denoted by $C^*$.

\[
C^* = \begin{pmatrix}
0.1428 & 0.1428 & 0.1428 & 0.1428 & 0.1428 & 0.1428 & 0.1428 \\
0.1428 & 0.0000 & 0.1428 & 0.1428 & 0.1428 & 0.1428 & 0.1428 \\
0.1428 & 0.2857 & 0.1428 & 0.2857 & 0.1428 & 0.1428 & 0.1428 \\
0.1428 & 0.1428 & 0.1428 & 0.1428 & 0.1428 & 0.1428 & 0.1428 \\
0.1428 & 0.1428 & 0.1428 & 0.1428 & 0.1428 & 0.1428 & 0.1428 \\
0.1428 & 0.1428 & 0.1428 & 0.1428 & 0.1428 & 0.1428 & 0.1428 \\
0.1428 & 0.1428 & 0.1428 & 0.1428 & 0.1428 & 0.1428 & 0.1428 \\
0.1428 & 0.1428 & 0.1428 & 0.1428 & 0.1428 & 0.1428 & 0.1428
\end{pmatrix}
\]

Figure 9: Sample shapes used in experiment.

To perform the experiment for sample shapes, we only need to compute the $p_j^{(0)}$, $j = 1, \ldots, n$, and run the PRS with each $p_j^{(0)}$ as initial input and repeat the process until it arrives at a stable state, i.e., $p_j^{(0)}$ converges to an unit vector. In our experiments, the features spaces of objects are obtained by the CBF method [16], and the PRS arrives at a stable state within 30 iterations. The sample shapes are shown in Fig. 8, and all the shapes are classified correctly (see Table 3).

Table 3: The FCMVs of samples (Fig. 8) with $C^*$

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
</tr>
</thead>
<tbody>
<tr>
<td>$p_1^*$</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>1.00</td>
</tr>
<tr>
<td>$p_2^*$</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>1.00</td>
</tr>
<tr>
<td>$p_3^*$</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>1.00</td>
</tr>
<tr>
<td>$p_4^*$</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>1.00</td>
</tr>
<tr>
<td>$p_5^*$</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>1.00</td>
</tr>
<tr>
<td>$p_6^*$</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>1.00</td>
</tr>
<tr>
<td>$p_7^*$</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>1.00</td>
</tr>
<tr>
<td>$p_8^*$</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>1.00</td>
</tr>
<tr>
<td>$p_9^*$</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>1.00</td>
</tr>
<tr>
<td>$p_{10}^*$</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>1.00</td>
</tr>
</tbody>
</table>

5 Conclusion

(i) PRS is simpler and also more abstracted than HAN and others. It can be used to classify any
kind of ‘objects’, *shapes, curves, time series, patients, students, equipment, market situations,* and so forth.

(ii) Competition is the essence of a PRS. It seems that the competition phenomena exists universally in the natural world and human society.

References


Evaluating the Usability of Visual Formalisms

Ghassan Al-Qaimari

Department of Computer Science, RMIT, Melbourne, Victoria, Australia
ghassan@cs.rmit.edu.au

ABSTRACT The design of user interfaces for information-intensive applications is a challenging problem, where the organisation, visualisation and manipulation of complex data are important. Visual formalisms are proposed as a good basis for the design of interfaces to complex scientific, engineering and business applications. They are diagrammatic displays with well-defined semantics for expressing relations.

This research work focuses on evaluating the usability of visual formalisms. We show by way of an example how to empirically assess the effectiveness of visual formalisms supported in an object-oriented database.

KEYWORDS Usability, visual formalisms, GUI, OODB, modelling constructs, evaluation

1. Introduction

Graphical user interfaces (GUI) with expressive visualisations are crucial to the success of underlying systems that support increasingly rich semantics. Mental models and metaphors have been proposed as bases for user interface design. Researchers who advocate mental models argue that interface design should closely match the way a user thinks of a task, and that interfaces should reflect people's mental models [HHW84]. On the other hand, other researchers believe that familiar, everyday metaphors such as desktops should be the starting point in interface design, since they can be interpreted by users based on prior knowledge of the source of the metaphor [CMK88]. Nardi and Zarmer [NZ93], however, argue that neither mental models nor metaphors provide a good basis for the design of interfaces to complex scientific, engineering and business applications. They claim that mental models are impractical, difficult to discover, and provide confusing design guidelines for the developer. While they accept that metaphors are useful for some purposes, they believe that metaphors lack precision and can lead to confusion due to uncertainty as to whether parts of the user's prior knowledge are applicable. Instead, Nardi and Zarmer [NZ93] propose visual formalisms as a superior basis for interface design. Visual formalisms are defined as "diagrammatic displays with well-defined semantics for expressing relations. They are based on simple visual objects such as tables, graphs, plots, panels and maps – objects that contain their own semantics and do not metaphorically recreate the semantics of some other domain" [NZ93]. According to Harel [Har88]: "The intricate nature of a variety of computer-related systems and situations can, and in our opinion should, be represented by visual formalisms: visual, because they are to be generated, comprehended, and communicated by humans; and formal, because they are to be manipulated, maintained, and analysed by computers."

In this research work, we describe one type of semantic data modelling constructs, that is composite objects. We use graph as well as form-based visual representations to capture the meaning of a composite object, its structural semantics and its relations to other objects. We introduce the concept of "lucidity" to describe the usability of the visual formalisms, and we summarise the results of our evaluation strategies to show how the concept of lucidity was used to guide the design of experiments to assess the effectiveness of the visual formalisms.

2. Background

Semantic data models and object-oriented databases (OODBs) have been proposed, in part, to overcome the limited modelling facilities of the relational model. However, such models are necessarily more complex than their relational predecessors, and thus are less readily associated with visual representations which are suitable for the wide range of tasks (data entry, querying, browsing, schema design, etc) associated with a database system. This problem is further exacerbated in some recent database systems that support sophisticated semantic data modelling constructs.

The example visualisations of modelling constructs presented in this paper were developed within the EVE (Extensible Visual Environment) direct-manipulation interface [PAK92] to the ADAM OODB [GKP92]. The EVE interface (shown in figure 1) supports visualisations
of a range of sophisticated data modelling constructs, such as relationship objects, versionable objects, composite objects and active rules.

3. Example - visualisations of composite objects

A number of objects related by the subpart relationship are collectively called a composite object [KBG89]. Each composite object has one or more subparts, each of which may consist of other composite objects, or standard ADAM objects. The semantics of the part-of relationship are discussed in [PDB93].

The following example defines a vehicle composite class:

```
composite class vehicle
attributes:
  registration_number: integer;
  owner: obj person;
composite attributes:
  door: obj door, set, excl, indep;
  engine: obj engine, single, excl, indep;
  wheel: obj wheel, set, excl, indep;
end;
```

In seeking lucid representations of composite object constructs, one approach which seemed promising was to investigate the application of alternative visual paradigms. Candidate paradigms which immediately suggested themselves were form-based and graph-based. Accordingly, three alternative representations of composite objects were designed. The form-based representation (figure 2) is very similar to EVE’s conventional way of visualising class instances (figure 1), with the exception that two different types of attribute are distinguished, namely composite (subparts) and non-composite.

In figure 3 the subpart hierarchy is depicted as a tree in a separate window. Note that the system defined relation subpart represents a special type of relationship between a composite object class and other classes. Thus the subparts window, shown in figure 3, could be eliminated if, for example, we chose to use a different line style from the one shown in the layout window (figure 1) to represent system-defined relationships. This would enable the user to differentiate between user-defined relationships and the subpart relationship. The justification for designing a subpart window is similar to the justification for having separate is-a hierarchy window, namely to highlight the special semantics of the relationship, and to prevent cluttering of the main layout window.

The representation, shown in figure 4, has all the characteristics of the form-based representation except for the subparts, which are shown as icons under the heading...
**Composite Attributes (Subparts).** As in the graphical representation, the user can click on any of the icons in the form to browse the subpart instances.

4. **Properties of visualisations**

Visualising information, especially complex and intricate information, has been the subject of considerable research, but little of this has related to the external representation of complex data models, and in particular modern object-oriented database systems.

The fundamental hypothesis underlying the work reported here is that the usability of object-oriented semantic data modelling constructs can be enhanced by lucid and expressive visual screen representations. Effective visual representations are of particular importance where modelling constructs are intended to capture both the structural and the behavioural semantics of real-world concepts. The challenge for the interface designer is to find clear, concise and comprehensive representations which achieve in practice the usability gains over textual representations which our hypothesis suggests are possible. Designing visual representations of data models involves a kind of **projection**, where the knowledge provided by the data model is transformed (mapped) into recognisable and expressive pictorial representations. This process has parallels with knowledge elicitation, as used by knowledge engineers in building knowledge-based systems [McG92]. Knowledge elicitation techniques described in the literature include: document reviews and content analysis, observation, interviews, concept and vocabulary analysis, and job and task analysis.

In [Nor88], the term **visibility** is used to indicate “the mapping between intended actions and actual operations”. It is also suggested that there are occasions when too much visibility can be a problem: “It is an excess of visibility that makes gadget-ridden, feature-laden modern audio sets and VCRs so intimidating.” The author also distinguishes between **affordance**, referring to whether the design of an object suggests (that is, affords) its functionality, and **perceived affordance**, referring to what a person thinks can be done with that object.

According to [Gil91], visibility should be thought of in terms of three separate dimensions, two of which are static properties of the presentation, while the third is dynamic. The first of the static components, **accessibility**, is associated with availability of information, while the other, **salience**, is associated with its meaning. Accessibility and salience are both properties of the display alone, independent of its use. The third dimension, namely **congruence**, is a property of the display in interaction with the user, and reflects whether the salience of a display is relevant to the user’s task. In a complementary analysis [JDM91], three distinct components of usability are investigated, which account for how the performance of a system changes with learning. These are: **guessability, learnability, and experienced user performance**. Guessability is defined as “a measure of time and effort required to get going with a system. The less time and effort required the higher the guessability.”

Through analysis of this work and its relevance to the derivation of visual representations for semantic data modelling constructs in object-oriented databases, we have been led to propose (in the next section) an additional usability dimension, referred to as **lucidity**. This concept incorporates aspects of **visibility** and **guessability**, as defined earlier, but we believe it is more directly characterises the effectiveness of a visualisation of a modelling construct for use in a database interface.
5. Lucidity of Visual Formalisms

In [Eas84], the usability of an interactive system is defined in general as "the extent to which the user can exploit the potential utility of a system." In other words, usability refers to how well users can use the functionality of an interactive system. Usability is traditionally associated with the following attributes [Shr92]: ease of learning; high speed of user task performance; user retention over time; low user error rate; and subjective user satisfaction.

Based on our practical experience in evaluating the usability of modelling constructs, we believe that a more specific definition that complements the above definition of usability and captures the special characteristics of visual formalisms is needed to guide usability specialists when they attempt to assess the effectiveness of their visualisations.

In addition to being structural and relational in nature, our visual formalisms have meanings of their own. In other words, they do not only bring with them their own semantics (as is the case in pie charts or hierarchies [Har88]), but rather attempt (through processes such as projection, knowledge distillation, and organisation of visual components), to reflect the semantics of the data model itself, and convey its meaning. This distinction adds to our visualisations of modelling constructs as an element of subjectivity, which is an important factor behind our attempt to devise a strategy for evaluating the usability of visual representations.

Lucidity, in the context of visualisation of modelling constructs, is defined as the extent to which a representation reflects and reveals the meaning and the structure of the underlying data model.

With this definition in mind, designers should aim at maximising the lucidity of visual representations. The different evaluation techniques which are conducted during the design and development process should be devised to measure whether the characteristics of a data model are visible to the user, and whether it is easy for the user to guess the structural semantics of the data model, and the function it is intended to perform. Lucidity may be considered to have the following attributes: clarity, referring to the number and organisation of visual elements on the screen; accessibility, referring to the ease of which information can be accessed; perspicuousness, referring to whether the visual formalisms convey to the user the appropriate meaning, and to whether they can be easily understood without confusion; and transparency, referring to whether users are able to see through the pictorial representation to the underlying semantics. Clarity and accessibility are directly related to the speed of performance, while perspicuousness and transparency are directly related to how much of the meaning and the structure of the underlying modelling construct is revealed.

6. Evaluation of Visual Formalisms

The evaluation of our modelling constructs is described in detail in [AKP95, APK94]. In this section we attempt to explain how the concept of lucidity can help the usability specialist in his/her endeavour to design evaluation tasks to measure the usability of visual representations.

6.1 Heuristic vs. formative evaluation

The goal of arriving at lucid representations of advanced semantic modelling constructs cannot be achieved just by following design guidelines, such as those in [MS86] and [Bro88]. An iterative approach must be adopted involving formative evaluation in each iteration.
[HH93].

Based on our experience, at least two evaluation cycles are needed to ensure that the visual representations capture the meaning and the structure of the underlying model. Firstly, a pre-implementation paper-based heuristic evaluation is highly recommended. Our evaluation was conducted with the help of expert users in the fields of databases and interface design. Secondly, a practical (task-oriented) evaluation was conducted using prototype implementations that incorporate the results of the first stage of the evaluation. The second technique is known as cooperative evaluation [MWH93]. It brings together designers and users in a cooperative context, so as to involve the users in the design process by giving feedback and identifying weak points at each stage.

The different backgrounds, expertise and experience of the representative users can provide valuable insights when the results of the evaluation are analysed and interpreted. Our subject in the first stage of the evaluation were all experts with considerable knowledge in the area of databases, and in the area of user interface design and development. Expert users can provide critical facts and heuristics at the early stage of the design. The subject profiles in the second stage of the evaluation (25 users) included novice to intermediate users, who are likely to be the primary users of the system. Such users usually have different needs, expectations and understanding of terminology. They also tend to draw different conclusions from instructions provided by the interface [McG92].

The first stage of evaluation involved checklists, questionnaires answered by expert users, which were used to obtain feedback on paper mock-ups of visualisations. Such an evaluation technique offers a flexible strategy, whereby checklists can be modified according to the nature of the evaluation, for the purpose of reaching practical quantifiable evaluation results. This evaluation strategy followed a similar approach to that presented in [RJ89, SH92].

Following consistent guidelines in the early stages of interface design ensures a consistent look and feel of the system as far as fonts, placement of menus, and wording of titles and messages are concerned. In addition to the design guidelines, the pre-implementation evaluation aimed to explore the extent to which the initial alternative visualisations succeed in capturing the structure and meaning of the data modelling constructs. This was achieved by encouraging the expert representative users to explore all initial designs of visualisations, to give constructive critiques, and to suggest possible alternatives. The first stage of the evaluation helps the designer to avoid, as much as possible, implementing poor visual representations that might require many revisions and modifications later at the prototyping phase. However, some ideas might seem very appealing when evaluated on paper, and yet due to unexpected limitations in the implementation environment, either cannot be implemented effectively, or turn out to be less effective in practice than anticipated.

In the second stage, we evaluated prototyped implementations of the proposed paper-based visual representations (see figures 2, 3, 4). All the prototypes were evaluated in parallel, and the type of formative evaluation conducted is usually known as exploratory evaluation, where users would attempt to perform representative tasks (or simply “walkthrough” the prototypes) and answer questions under the guidance of a test monitor. The representative users were asked to perform certain specific tasks, and while looking at the information available to them on screen, they were asked post-experience questions the aim of which was to assess the usability of the system.
Your task is to find the *engine* number of the *engine* of the *vehicle* which has the *registration* number: 1234.

After carrying out the task, use the information available to you on the screen to answer to each of the following post-experience questions.

1. Who is the owner of this vehicle? How did you know?
2. How many vehicles are stored in the database?
   How did you know?
3. How many doors does the vehicle have?
   How did you know?
4. Can two vehicles share the same engine?
   Why/Why not?
5. What will happen to the vehicle if you try to remove an *owner* from the database?
   How did you know?
6. What fundamental difference is there in the way the *registration* and the *engine* are modelled as properties of vehicle?
   How did you know?
7. What is the fundamental difference between the way in which the *engine* of a vehicle is modelled compared with the *owner* of a vehicle?
   How did you know?
8. Is an engine a normal object in the database?
   Why do you think it is/Why do you think it is not?
9. Is it a good idea to have a special button in the dialog box (figure 2 and 4) which gives the user the choice of whether to display the *Subparts Hierarchy Window* (figure 3) or not.
   Explain why it is/isn’t a good idea?

Figure 5: A representative evaluation task followed by post-experience questions.

in general, and the lucidity of the each visual representations in particular (see figure 5).

### 6.2 Designing an evaluation task

Designers and evaluators use different evaluation methods to achieve an objective measurement of the user/system interaction, and yet it is in fact the subjectivity of the evaluation experience of the individual users which they are often after [CHP90].

As indicated earlier, one of our aims was to measure the *lucidity* of the visual representations of the data modelling constructs supported in the EVE interface, in addition to assessing the usability of the system in general. To do so, it was necessary to devise an evaluation task that reflects what is meant by lucidity and how it could be measured.

The tasks performed by our representative users in the second stage of the evaluation aimed at exploring the following interface issues:

- Do the visual representations succeed in suggesting to the user the meaning of the data model, and do they reveal the semantics supported by the modelling constructs?
  - Is it clear to the user that a construct, such as a composite object, is a normal object in the database?
  - How useful are icons for quick identification of related visual representations?
  - How helpful is a hierarchical overview of special types of relationships?

It was equally important after deciding on the tasks to appraise them in order to check their representativeness and the extent to which they explore the interface issues under evaluation. To do so, the proposed tasks were carried out by an experienced user in a pilot investigation.

During evaluation sessions the emphasis was placed on the reason behind *how* the user found a certain answer, and *why* a certain conclusion was reached or a decision made. Furthermore, an emphasis was placed on finding out whether the representative users performed subtasks within a reasonable period of time, which was an important factor in determining the effectiveness of a visual representation. Users were encouraged to to think aloud, in order to generate as much feedback as possible. This
was achieved by asking questions after performing each subtask (such as How did you know that? or What makes you think so?, etc.). For example, by asking the representative user a question like: How many vehicles are stored in the database?, the designer is actually interested in finding out whether the counter in the representation is visible to the user, and whether or not the wording is confusing.

The representative users were kept unaware of the fact that the time taken to perform each subtask was being monitored. Such an informal evaluation strategy created a relaxed atmosphere, and enabled representative users to concentrate better on the screen without interruption.

While the representative user is carrying out the task, the test monitor records feedback information on a think aloud protocol recording sheet. Such a recording sheet contains questions such as: What does the task user notice? What is the task user thinking now? What kind of clues is he/she looking for? What are the problems encountered at this subtask? What suggestions are being made by the task user?

In general, the first two attributes of lucidity, namely clarity and accessibility, are directly related to the speed of performance, therefore, the evaluation tasks should measure how long it takes the user to complete the task successfully. The third and fourth attributes of lucidity, namely perspicaciousness and transparency, are directly related to the strategy. In other words, if the visual formalism is poorly designed and the user fails to understand its meaning or its structure, then the user’s next step is likely to be incorrect. The user’s feedback is crucial to assessing the perspicaciousness and transparency of visual formalisms, and the evaluation task should be designed to allow the test monitor to elicit the user’s mental model. Exploratory evaluations were suitable for assessing the perspicaciousness and transparency of the visualisations because they tend to be informal and almost collaboration between participants and test monitors, with much interaction between the two. Unlike clarity and accessibility where the emphasis is on measuring how well the user is able to perform (quantitative data), exploratory evaluations tend to solicit users’ ideas about how to improve confusing areas (qualitative data).

7. Conclusion and future directions

This paper attempted to exploit another dimension in interface evaluation, that is, the degree of lucidity of visual representations, and has demonstrated how the attributes of lucidity influence the types of experiments designed to assess the effectiveness of visualisations.

The need for expressive visual representations becomes important as the underlying systems support increasingly rich semantics. Our experience proved that it is possible to make advanced data modelling constructs accessible to non-specialist users.

Evaluation is crucial to the development of successful interfaces with effective and lucid visualisations. We believe that cooperative evaluations play a crucial role in bringing the designers and the users together in a context that involves the users in every step of the iterative design process.

References


Comparative Study of Weighted Median Filters

WILLIAM T.M. DUNSMUIR and YUAN ZOU
University of New South Wales
Sydney NSW 2052, Australia
November 21, 1996

Abstract

In this paper, we will derive the output distribution of weighted median filtering with independent but not identical inputs. Then we will propose using exponential weights for weighted median filtering and study the noise suppression, edge and line preservation ability of weighted median filtering for different weights and window shapes. Computational results are given to show that weighted median filtering can preserve image details while suppressing noise.

1 Introduction

In the last few decades, many types of nonlinear filters based on order statistics have been introduced. One of the most well known is median filtering. The one dimensional median filter was first devised by Tukey[12] in time series analysis. Since then, median filters have been widely used in one and two dimensional signal and image processing. Pratt[9] first applied the median filtering to image noise smoothing. He used the median intensity in the moving window to replace the window's central pixel and concluded that the technique is effective for removing impulsive noise from image while preserving the edges. Some other applications can be found in [1] and [2]. The weighted median filtering is an extension of the median filtering and has also the properties of edge preservation and noise suppression. It was first introduced by Justasson[6] and further discussed in Brownrigg[3]. Compared to the median filtering, it is only possible to adjust the window shape and size, the weighted median filtering can choose different weights so that it is potentially more flexible in preserving important signals and suppressing impulsive noise than median filtering.

Ko and Lee[7] considered the center weighted median filter, which giving more weight only to the central value of a window is easier to implement than the general weighted median filters. Huang et al.[4] proposed to use exponential weights for weighted median filtering in image processing. Dunsmuir et al.[5] derived the output distribution of exponentially weighted moving median for i.i.d. data for quality control application. Prasad et al.[8] used the rank selection probability and related concepts to analyze statistical properties of weighted median filters. Some statistical properties of weighted median filters using stacking property and threshold decomposition have been discussed by Yli-Harja et al. [11].

In this paper, we first define the weighted median filtering associated with real-valued weights. Some definitions and notations are given in section 2. The output distribution of weighted median filtering with independent but not identical inputs are derived in section 3. In section 4, we study the noise suppression and edge and line preservation ability of weighted median filtering for different weights and window shapes by statistical analysis. Conclusions are given in section 5.

2 Weighted median filters

Let $X = (X_1, X_2, \ldots, X_N)$ be a collection of independent identically distributed random variables having the same cumulative distribution function $F(x)$. Associate with the $X_i$ a weight $w_i$ and let $w = \{w_1, w_2, \ldots, w_N\}$. We assume $N = 2n + 1$.

Let $X(1) \leq X(2) \leq \cdots \leq X(N)$ be the order statistics of $X_1, X_2, \ldots, X_N$ and their associated weights be denoted by $w_{[1]}, w_{[2]}, \ldots, w_{[N]}$.

The weighted median filter is commonly defined as follows:

\begin{figure}[h]
\centering
\includegraphics[width=0.8\textwidth]{fig1.png}
\caption{Common filter windows for two-dimensional median filtering.}
\end{figure}
Definition 2.1 For $X$ and integer weights $w$, let the sum of the weights be an odd number. The output of the weighted median filter of span $N$ is given by:

$$Y = \text{med}\{w_1 \circ X_1, w_2 \circ X_2, \cdots, w_N \circ X_N\}$$ \hspace{1cm} (1)

where $\circ$ denotes duplication

$$n \circ X = \underbrace{X, \cdots, X}_{n \text{ times}}$$ \hspace{1cm} (2)

See [6] for example. From definition 1, we know that the output $Y = X_{(k)}$ if and only if:

$$\sum_{i=1}^{k-1} w_{i[i]} < \sum_{i=k}^{N} w_{i[i]}$$ \hspace{1cm} (3)

and

$$\sum_{i=1}^{k} w_{i[i]} \geq \sum_{i=k+1}^{N} w_{i[i]}$$ \hspace{1cm} (4)

The above can be naturally generalized to the case of positive real valued weights $w_i, i = 1, \cdots, N$. See [5] for example. Let $w^*_j = w_j/\sum_{i=1}^{N} w_i$ and define a weighted empirical distribution function

$$F_N(x) = \sum_{i=1}^{N} w^*_i I_{[X_i \leq x]}$$ \hspace{1cm} (5)

Definition 2.2 The output of the weighted median filter of $X$ with associated weights $w$ is denoted as $\hat{m}$ and defined by:

$$\hat{m} = \inf \{x \mid F_N(x) \geq 0.5\}$$ \hspace{1cm} (6)

The output of the weighted median filter with positive real-valued weights can be calculated as follows: the samples in the filter window are sorted in ascending order, and the corresponding weights from the smallest to the largest are added up until the partial sum is $\geq 0.5 \sum_{i=1}^{N} w_i$, then the ordered sample corresponding to the last weight added is the output the weighted median filter. When $w^*_i = 1/N$, the output is standard median filtering.

For the remainder of this paper, we will assume the weights are real, non-negative and sum to unity so that $w_i \geq 0$ and $\sum_{i=1}^{N} w_i = 1$. Using notation similar to that in [12], let the set of all combinations taking $i$ weights from $w$ be

$$\gamma^i = \{A \mid A \subseteq w, |A| = i\}, i = 1, \cdots, N$$ \hspace{1cm} (7)

where $|A|$ denotes cardinality the of the set $A$.

The set of all the combinations taking $i$ weights from $w$ such that the sum of the weights is greater than or equal to 0.5 is denoted by $\Omega^i$.

$$\Omega^i = \{A \mid A \subseteq \gamma^i, \sum_{w_j \in A} w_j \geq 0.5\}$$ \hspace{1cm} (8)

Let the number of ways of doing this be denoted by

$$M_i = |\Omega^i|, i = 1, \cdots, N.$$ \hspace{1cm} (9)

3 Output distributions of weighted median filtering

When we consider applications to edge detection, we need to assume that the filter window is $V_1 \cup V_2$ and $V_1 \cap V_2 = \emptyset$ with $|V_1| = m_1$ and $|V_2| = m_2 = N - m_1$. On $V_1$, $m_1$ observations $X_{11}, X_{12}, \cdots, X_{1m_1}$ with associated weights denoted by $w_1^{(1)}$ are drawn from $F_1(x)$ with density function $f_1(x)$. And there are $m_2 = N - m_1$ observations $X_{21}, X_{22}, \cdots, X_{2m_2}$ on $V_2$ with associated weights denoted by $w_2^{(2)}$ are drawn from $F_2(x)$ with density function $f_2(x)$. We define the empirical distribution of the combined samples of size $N$ as before. Let $X \leq X \leq \cdots \leq X_{1(N)}$ be the order statistics of the combined samples, and their corresponding weights be denoted by $w_{11}, w_{12}, \cdots, w_{1N}$. Let $X^{(1)} \leq X^{(2)} \leq \cdots \leq X^{(m_1)}$ be the order statistics on $V_1$ with the associated weights $w_{11}, w_{12}, \cdots, w_{1m_1}$, and $X^{(2)} \leq X^{(3)} \leq \cdots \leq X^{(m_2)}$ be the order statistics on $V_2$ and $w_{21}, w_{22}, \cdots, w_{2m_2}$ be the associated weights. Then the output distribution function of $\hat{m}$, denoted by $F_{\hat{m}}(x)$, is:
Theorem 3.1

\[ F_{\hat{m}}(x) = \sum_{i=1}^{N} \sum_{r=0}^{m_1} M_{r,i-r} F_1(x)^r \times (1 - F_1(x))^{m_1 - r} F_2(x)^i \times (1 - F_2(x))^{m_2 - i + r} \]

where

\[ \gamma_{[i]} = \{ A | A \subseteq w^{(1)}, |A| = i \}, \]

\[ \gamma_{[j]} = \{ B | B \subseteq w^{(2)}, |B| = j \}, \]

\[ \Omega_{ij} = \{ A \cup B | A \subseteq \gamma_{[i]}, B \subseteq \gamma_{[j]} \}, \]

\[ \sum_{w_i \in A} w_i + \sum_{w_j \in B} w_j \geq 0.5, \]

\[ i = 0, 1, \ldots, m_1; j = 0, 1, \ldots, m_2. \]

\[ \Omega_{ij}, \text{ and } M_{ij} \text{ denotes the cardinality of } \Omega_{ij}, \text{ i.e.} \]

\[ M_{ij} = |\Omega_{ij}|, \]

\[ i = 0, 1, \ldots, m_1; j = 0, 1, \ldots, m_2. \]

Proof:

\[ F_{\hat{m}}(x) = P(\tilde{m} \leq x) = P(\bigcup_{i=1}^{N} \{ \text{exactly i samples} \leq x \text{ and the sum of their associated weights} \geq 0.5 \}) \]

\[ = \sum_{i=1}^{N} P(X_{(i)} \leq x < X_{(i+1)}), \]

\[ \sum_{i=1}^{N} w_i I_{[x \leq x]} \geq 0.5 \]

\[ = \sum_{i=1}^{N} \sum_{r=0}^{m_1} P(X_{(r)} \leq x < X_{(r+1)}, x < X_{2(i-r+1)}, \sum_{i=1}^{N} w_i I_{[x \leq x]} \geq 0.5) \]

\[ = \sum_{i=1}^{N} \sum_{r=0}^{m_1} P(X_{(r)} \leq x < X_{(r+1)}, x < X_{2(i-r+1)}, \sum_{i=1}^{N} w_i I_{[x \leq x]} \geq 0.5) \]

The density function of the output of weighted median filter, denoted by \( p(x) \), is

\[ p(x) = \sum_{i=1}^{N} \sum_{r=0}^{m_1} M_{r,i-r} F_1(x)^{i+r} (1 - F_1(x))^{m_1 - r} \times (1 - F_2(x))^{m_2 - i + r} \]

Corollary 3.1 When the weights are equal, \( w_i = 1/N, i = 1, 2, \ldots, N \), we have

\[ F_{\hat{m}}(x) = F(x)^{i+r} (1 - F(x))^{m_1 - r} \times (1 - F_2(x))^{m_2 - i + r} \]

\[ = \sum_{i=1}^{N} \sum_{r=0}^{m_1} M_{r,i-r} F(x)^{i+r} (1 - F(x))^{m_1 - r} \times (1 - F_2(x))^{m_2 - i + r} \]

Since \( \sum_{r=0}^{m_1} M_{r,i-r} = M_i \), when \( F_1(x) = F_2(x) = F(x) \), we have

\[ F_{\hat{m}}(x) = \sum_{i=1}^{N} M_i F(x)^{i+r} (1 - F(x))^{m_1 - r} \times (1 - F_2(x))^{m_2 - i + r} \]

\[ = \sum_{i=1}^{N} M_i F(x)^{i+r} (1 - F(x))^{N-i} \]
So the output distribution function of \( \tilde{m} \) is:

**Theorem 3.3**

\[
F_{\tilde{m}}(x) = \sum_{i=1}^{N} M_i F(x)^i (1 - F(x))^{N-i}
\]

The result is the same as that in [10] using integer weights and [5] using real-valued weights.

4 Statistical Analysis of Weighted Median Filtering

Median filter has widely been used for noise suppression and edge preservation. When the window size is not big enough, the noise suppression ability for median filtering is poor. But when the window size is too big, median filtering will smear the edge. In this paper, we will extend the application of exponential weights used in [4] and [5]. Let \( W \) denote the filter window, \((i,j)\) denotes the centre of the window. We use exponential function to distribute the weights, i.e.

\[
w(i,j; k, l) = a(1 - a)^{d(i,j; k, l)}, \ (k, l) \in W
\]  

where

\[
d(i, j; k, l) = \sqrt{(k-i)^2 + (l-j)^2},
\]

\((k, l) \in W\)  

The exponential weights are a one parameter family which ranges from centre weights to equal weights as the parameter \( a \) goes from 1 to 0. The ultimate aim is how to choose the parameter to be locally adaptive to image feature and noise.

We will study the noise suppression and edge and line preservation ability using weighted median filters with different weights and window shapes. Statistical analysis will be used. First, we will calculate the output variances of the weighted median filtering when the input signals are constant signals embedded in additive white noise. In order to get some information about how the filters can preserve the line and edge in image, we will compute and compare the output mean value and root mean square errors to show how well the filters can preserve the details and suppress noise by considering \( 2-D \) input signals corrupted by additive white noise with step edges and lines. The following results were computed using numerical integration.

4.1 Noise Suppression

Using theorem 3.3, we compute the output variance of the weighted median filtering with standard normal distribution as the inputs, i.e., \( F(x) \) has a standard normal distribution function. Different window shapes and weights will be considered.

For the given weights \( a = 0.15, 0.35, 0.55, 0.75 \) and different window shape, we consider the window size \( N = 3, 5, 7, 9 \) for the bar and elbow windows, \( N = 9 \) for square, cross and X-shape window. The computational results are given below.

Figure 1 and 2 show that the median filter always has the smaller output variance compared to the weighted median filter with the same window shape and window size and the window size is bigger, the output variance is smaller. The output variance increases as the parameter \( a \) increases.

![Figure 2: Output variances of weighted median filters for bar window.](image)

The relationship between the output variances and the weights in the filter window are graphed in Figure 4, which demonstrates the output variances are proportional to the sum of square weights in the filter windows as might be expected by analogy with the variance of weighted average.
4.2 Edge Preservation

We consider the edge preservation ability of weighted median filtering and define our edge model as:

\[
z_{ij}^v = \begin{cases} N_{ij} & \text{if } j \leq 0 \\ h + N_{ij} & \text{if } j \geq 1 \end{cases}
\]

where \( h \) represents the edge height and \( N_{ij} \) are assumed to be i.i.d. noise with distribution \( F(x) \), which we assume is the standard normal distribution in what follows.

Under the edge model, using theorem 3.2 and letting \( F_1(x) = F(x) \) and \( F_2(x) = F(x - h) \), for the given weights \( a = 0.15, 0.35, 0.55, 0.75 \), we consider square window with window size 9 and compute the output expected values and root mean square errors (rmse) of weighted median filtering when the step edge \( h = 4 \). The results are given in Figure 5. For the cross and X-shape window, the computation results are very similar to the result of square window. The results show that the filters can preserve edge.

4.3 Line Preservation

We define the line models as follows:

\[
z_{ij}^v = \begin{cases} N_{ij} & \text{if } j \neq 0 \\ h + N_{ij} & \text{if } j = 0 \end{cases}
\]

where \( h \) is the height of the line and \( N_{ij} \) are assumed to be i.i.d. noise with distribution \( F(x) \).
In the following, we assume that it is standard normal distribution.

We consider the line preservation ability of weighted median filters by computing the output expected values and root mean square errors (rmse) of weighted median filtering. For the given weights $a = 0.15, 0.35, 0.55, 0.75$, we consider square window with window size 9 when the step edge $h = 4$. The computational results are given in Figure 7 and 8. For the cross and X-shape window, the computational results are very similar to the result of using square window. The results show that median filter removes the line and the line preservation ability is better as the parameter $a$ increases.

5 Conclusions

We derived the output distribution function of the weighted median filtering with independent but not identically distributed inputs. Then, we proposed using exponential weights and compute the variance, expected values and root mean square errors of the output of weighted median filtering. The computation results show that weighted median filter can preserve image details better than the median filter. The results presented in this paper give us some criteria to choose the parameter $a$.

References


Automatic Human Face Detection and Recognition under Non-uniform Illumination

Toshiaki Kondo* and Hong Yan

Department of Electrical Engineering
University of Sydney
N.S.W. 2006
Australia
Email: kondo@ee.usyd.edu.au

Abstract

A system for automatic human face detection and recognition is presented. The procedure consists of five steps: (1) Haar wavelet transform, (2) facial edge detection, (3) symmetry axis detection, (4) face detection by template matching and (5) face recognition based on principal component analysis (PCA). Face detection is performed at a low level of the input image for an efficient search. PCA-based face recognition identifies the face after the illumination is normalized. Our method shows a robust performance for images taken with a cluttered background under non-uniform illumination.

1 Introduction

Human face detection and recognition have drawn considerable interest and attention from many researchers for decades [1], [2]. However, it is still a challenging research topic since the human face may change its appearance because of facial expressions, beard, mustache, hair styles, make-up, glasses, aging, surgery etc. In addition to these internal variations, we have to consider the external distortions such as the scale, lighting, position, tilt and orientation of the face. It should be also noted that a complex background in an image makes it far more difficult to locate faces and facial features. Therefore, it is always necessary for an automatic face recognition system to set some constraints on input images. Our goal is to relax these constraints. In this paper, we focus on images captured with a cluttered background under non-uniform illumination.

2 Face Detection

Figure 1 shows the overall flow diagram of our system. We describe steps 1 to 4 in this section. These procedures aim to realize a system which allows users to stand rather freely in front of a camera with a cluttered background.

2.1 Haar Wavelet Transform

A multifrequency channel analysis is employed for an efficient search of a face. In our experiment, the wavelet transform decomposes the input image \( f(x, y) \) with a resolution of 256 by 240 pixels with 256 gray levels into four subimages of size 128 by 120 pixels. The Haar function is used as a wavelet since it is the simplest linear phase filter of all wavelets. The four subimages are a low pass filtered image \( f_{LL} \), a horizontally low pass filtered and vertically high pass filtered image \( f_{HL} \), a horizontally high pass filtered and vertically low pass filtered image \( f_{HL} \) and a high pass filtered image \( f_{HH} \). As shown in Figure 1, subimage \( f_{LL} \) is used for face detection, while both \( f_{LH} \) and \( f_{HL} \) are used in facial edge detection and symmetry axis detection. An input image and all its subimages except for \( f_{HH} \) are illustrated in Figure 5 (No.0, 1, 2 and 3). \( f_{HH} \) is not currently utilized as it does not contain much image information.

2.2 Facial Edge Detection

Facial edge detection can be divided into the three parts as shown in the dash boxes in Figure 1. Oriented edges are effectively detected from subimages \( f_{LH} \) and \( f_{HL} \). The horizontal edges surrounding the eyes and mouth are extracted from \( f_{LH} \), whereas the vertical edges of the nose are detected in \( f_{HL} \). Then the binary edge image is obtained by the logical OR operation between the two subimages \( f_{LH} \) and \( f_{HL} \) after they are converted to binary images with a prespecified threshold \((\geq 1 \pm 4)\).

Next, the edge image is partitioned into small blocks of size 8 by 8 pixels (see No.4 in Figure 5) and the number of edges in each block is determined. A block containing a sufficient number of edges \((\geq 8/64)\) is selected as an edge block, while a block with few edges is ignored in the following process as a plain background (see No.5 in Figure 5).

Finally, we exclude the non-facial edge blocks by examining their connections with neighboring blocks. We regard a block as valid which has adjoining edge blocks on both sides and in its downward direction. This T-
and $f_{IH}$. Subimages $f_{HL}$ and $f_{IH}$ are equivalent to the image gradients described in the following equations respectively:

$$f_{HL} = \frac{\partial f(x,y)}{\partial x}, \quad f_{IH} = \frac{\partial f(x,y)}{\partial y}. \quad (1)$$

Figure 3 illustrates the gradient vectors. Assume the image in the right half is brighter than that in the left half. As the length of each vector changes according to the illumination on it, the gradient vectors in the right half are longer than those in the left. In this way, all intensity or gradient based methods are influenced by changes in lighting conditions.

We use the gradient orientation $\theta(x,y)$ to detect the symmetry axis because we have found it is illumination-invariant. Even if the face is shadowed, the gradient orientation is preserved except for the border of the shadow. The gradient orientation is defined as

$$\theta(x,y) = \arctan \frac{f_{IH}}{f_{HL}}. \quad (2)$$

The range of $\theta$ is $[0, 2\pi]$. Equation 2 can be computed efficiently provided a lookup table is used.

A block assumed to have roughly the same size, $h$ by $v$, as that of the expected input face is scanned over the facial edge area for computing the symmetry index $s_{xy}$. The definition of $s_{xy}$ is given by

$$s_{xy} = \sum \sum \left\{ |\cos \theta(i,j) + \cos \theta(2x - i,j)|^2 ight. \\
\left. + |\sin \theta(i,j) - \sin \theta(2x - i,j)|^2 \right\}, \quad (3)$$

where $(y - v/2) \leq j \leq (y + v/2)$ and $(x - h/2) \leq i \leq x$.

When the image in the block is bilaterally symmetric, each term of Equation 3 takes a small value because of cancellation. On the contrary, when it is asymmetric, the index gives a larger value. This can be seen in Figure 5 (No.7). The dark vertical stripe shows the symmetry axis centered on the face. The next process is to be carried out over the high symmetry index area shown in Figure 5 (No.8 and 9).
2.4 Template Matching

Normalized cross correlation is computed between the input image \( f(x, y) \) and a face template \( w(x, y) \) shown in Figure 4. The definition is given as

\[
\gamma_{xy} = \frac{\sum_{i,j} [f(i, j) - \overline{f}][w(i - x, j - y) - \overline{w}]}{\sqrt{\sum_{i,j} [f(i, j) - \overline{f}]^2 \sum_{i,j} [w(i - x, j - y) - \overline{w}]^2}},
\]

where \( \overline{f} \) is the average of \( f(x, y) \) over the overlapped region with \( w(x, y) \), and \( \overline{w} \) is the average of \( w(x, y) \). The correlation coefficient \( \gamma_{xy} \) is scaled in the range -1 to 1. This computation is carried out separately in the left and right half of the template to reduce the effect of the non-uniform illumination. The face template of size 40 by 48 pixels is obtained simply by taking the average of all face images manually cropped from the database produced by the MIT Media Lab. Again, by eliminating the locations with low correlation coefficients (\( \gamma_{xy} \leq 0.18 \)), the face-like area is extracted from the possible face area which has passed the preceding steps (see No.10 in Figure 5).

![Figure 4: Face template.](image)

2.5 Results

We have tested our face detection scheme under three different lighting conditions using 45 images from 15 persons. The best matching point in Figure 5 (No.11) shows the face location detected by our method, and the face template is superposed on \( f_{LL} \) in No.12.

The face detection rates are given in Table 1. Our algorithm shows a robust performance under non-uniform illumination. The only detection error occurred when the only Oriental face in the database was used. The face template needs to be similar enough to each face in the database. Hence, we should have several different face templates when the number of face images increases. Alternatively, it would be effective to adopt a deformable template matching strategy [4].

<table>
<thead>
<tr>
<th>Illumination angle</th>
<th>Detection rates</th>
</tr>
</thead>
<tbody>
<tr>
<td>Head-on</td>
<td>15/15 100.00%</td>
</tr>
<tr>
<td>45 degrees</td>
<td>14/15 93.33%</td>
</tr>
<tr>
<td>90 degrees</td>
<td>15/15 100.00%</td>
</tr>
</tbody>
</table>

Table 1: Face detection results.

![Figure 5: Procedures of face detection.](image)

3 Face Recognition

Our face recognition method is based on principal component analysis (PCA) since the principal component of the K-L expansion is the optimal filter that maximizes the signal-to-noise ratio of the output correlation when the input signal is distorted [5].

3.1 Principal Component Analysis

At the recognition stage, we use the full scale face image of size 80 by 96 pixels instead of the low level face image to achieve higher accuracy.

3.1.1 Computation of Eigenfaces

Let us summarize the computation of eigenfaces [6], [7]. Assume a set of face images to be \( \{ \Gamma_1, \Gamma_2, \ldots, \Gamma_M \} \). The average face of the set is defined by \( \Psi = \frac{1}{M} \sum_{k=1}^{M} \Gamma_k \), which is identical to the face template mentioned before. Each face differs from the average by the vector \( \Phi_k = \Gamma_k - \Psi \).
We define the matrix $A$ as $[\Phi_1 \Phi_2 \cdots \Phi_M]$. The matrix $A$ has $M$ columns and $N$ rows. The number $N$ is equal to the number of pixels in an image. Consider the eigenvectors $u_i$ of $AA^T$ and $v_i$ of $A^TA$ such that

$$
(AA^T)u_i = \mu_i u_i, \quad (A^TA)v_i = \mu_i v_i,
$$

where $\mu_i$ are the eigenvalues. It is always possible to diagonalize the matrix $AA^T$ and $A^TA$ because they are real and symmetric. However, it is laborious to solve Equation 5 directly since the number $N$ is huge. A mathematical technique is used here. Multiplying both sides of Equation 6 by $A$, we obtain

$$
(AA^T)(Av_i) = \mu_i (Av_i).
$$

Comparing Equation 5 with Equation 7, we have

$$
u_i = Av_i.
$$

Thus, we only have to solve Equation 6 to obtain $v_i$. The eigenvectors $u_i$ are referred to as eigenfaces or eigencolors as they appear face-like. Figure 6 shows the first eigenface of size 80 by 96 pixels.

It is known that the original image $\Phi_k$ can be represented as a weighted sum of eigenvectors $u_i$ as follows:

$$
\Phi_k = \sum_{i=1}^{M} u_i (u_i^T \Phi_k)
$$

The scalars $u_i^T \Phi_k$ determine the weights of eigenvectors $u_i$. The eigenfaces and weights are computed when the database is established and they are made available during the recognition process. The weights of the input image $\Phi_n$ are also obtained by the inner product $u_i^T \Phi_n$.

We discriminate individuals using the distribution of the weights $u_i^T \Phi_n$.

Figure 6: First eigenface $u_1$.

### 3.1.2 Recognition Process

PCA-based recognition is fairly simple and fast in comparison with feature-based methods. The input face image $\Gamma$ is projected into the eigenface space after the average face $\Psi$ is subtracted. The projection is simply an inner product between the input face and each eigenface:

$$
\omega_i = u_i^T (\Gamma - \Psi),
$$

for $i = 1, 2, \ldots, M$. The weights $\omega_i$ form a vector $\Omega^T = [\omega_1, \omega_2, \cdots, \omega_M]$ that shows the contribution of each eigenface. The input face is identified by finding the minimum Euclidean distance $\epsilon$ defined as

$$
\epsilon_i = ||(\Omega - \Omega_i)||^2,
$$

where $\Omega_i$ is the precalculated vector of the $i$th face in the database.

### 3.2 Normalized PCA

We discuss the relation between PCA-based face recognition and illumination. The MIT research group claimed that the PCA-based algorithm was robust to variations in lighting conditions [7]. In fact, the face recognition ratio was not strongly affected by non-uniform illumination in their report. However, there is a crucial problem in their experiment. Although they multiplied the input face image by a two-dimensional gaussian window centered on the face, the background is not completely excluded. What is worse, each face has its peculiar background which is fixed under the different lighting conditions. Since each background is a fixed pattern attached to each face, it helps to discriminate individuals. This problem is also mentioned in [8].

We conducted a similar simulation to confirm this positive effect of the background, especially under different illumination conditions. To make it clear, we did not employ any filter for suppressing the background. All 80 by 96 face images are cropped from the full scale images manually. Table 2 shows the recognition results of PCA using the full scale images, whereas Table 3 shows the results when only the face images are used.

<table>
<thead>
<tr>
<th>Illumination angle</th>
<th>Recognition rates</th>
</tr>
</thead>
<tbody>
<tr>
<td>head-on</td>
<td>15/15 100.00%</td>
</tr>
<tr>
<td>45 degrees</td>
<td>15/15 100.00%</td>
</tr>
<tr>
<td>90 degrees</td>
<td>14/15 93.33%</td>
</tr>
</tbody>
</table>

Table 2: PCA using 256x240 full scale images.

<table>
<thead>
<tr>
<th>Illumination angle</th>
<th>Recognition rates</th>
</tr>
</thead>
<tbody>
<tr>
<td>head-on</td>
<td>15/15 100.00%</td>
</tr>
<tr>
<td>45 degrees</td>
<td>6/15 40.00%</td>
</tr>
<tr>
<td>90 degrees</td>
<td>10/15 66.67%</td>
</tr>
</tbody>
</table>

Table 3: PCA using 80x96 face images.

The stark contrast between Table 2 and Table 3 shows that the background positively contributes to the recognition process. As a result, we should address that PCA-based recognition is rather sensitive to changes in lighting conditions and preprocessing should be considered.

In our method, we conduct illumination normalization before PCA. We have normalized all face images cropped from the database before computing eigenfaces. The normalization is carried out over the entire area of each face.
since the face in the database should have been illuminated uniformly. In the recognition process, the input face image is also normalized before being projected into the eigenspace. This normalization is performed separately in the left and right half of the face to reduce the effect of the non-uniform illumination.

3.3 Results
The result of face recognition by normalized PCA is shown in Table 4. The recognition rates have been successfully improved under oblique illumination (45°). On the other hand, we could not achieve higher recognition rates under side illumination (90°). However, there is a significant improvement in reliability in spite of the same recognition rates. We define the reliability index as

\[
R = \frac{\epsilon_{\text{end}} - \epsilon_{\text{min}}}{\epsilon_{\text{min}}},
\]

where \(\epsilon_{\text{min}}\) and \(\epsilon_{\text{end}}\) are the Euclidean distances of the best and the second best matching. The larger the index \(R\), the more reliable the recognition result. The reliability indices are given in Table 5. The reliability leaps by 3.5 times after illumination normalization in both lighting conditions.

<table>
<thead>
<tr>
<th>Illumination angle</th>
<th>Recognition rates</th>
</tr>
</thead>
<tbody>
<tr>
<td>head-on</td>
<td>15/15</td>
</tr>
<tr>
<td>45 degrees</td>
<td>14/15</td>
</tr>
<tr>
<td>90 degrees</td>
<td>10/15</td>
</tr>
</tbody>
</table>

Table 4: Normalized PCA using 80x96 face images.

<table>
<thead>
<tr>
<th>Illumination angle</th>
<th>PCA</th>
<th>Normalized PCA</th>
</tr>
</thead>
<tbody>
<tr>
<td>45 degrees</td>
<td>1.425550</td>
<td>5.071897</td>
</tr>
<tr>
<td>90 degrees</td>
<td>0.479777</td>
<td>1.696856</td>
</tr>
</tbody>
</table>

Table 5: Average reliability of correct recognition results.

3.4 System Integration
We discuss how face detection and recognition are integrated here. System integration is an essential part because we carry out face detection in a lower resolution. Figure 7 depicts the flow of the integration. First of all, the best matching point determined by our face detection scheme is projected into the original input image of size 256 by 240. Normalized PCA is applied to the project pixel and its eight adjacent pixels, namely nine pixels altogether. The recognition result is determined by \(R_{\text{max}}\). If \(R_{\text{max}} \geq 1.0\), the result is regarded as reliable and the recognition process is terminated. If \(R_{\text{max}} < 1.0\), we apply normalized PCA to the other face candidates which satisfy \(\gamma_{xy} > 0.85 \times \gamma_{\text{max}}\). In this integration step, the only face detection error was corrected. The integrated system achieved the same results as shown in Table 4 automatically.

3.5 Evaluation
The important issue regarding face recognition is how to suppress a false acceptance rate (FAR) and false rejection rate (FRR). This is often more important than face recognition accuracy. We attempt to reduce these rates by setting a threshold against the reliability \(R\). Table 6 and Table 7 show recognition rates, FARs and FRRs at two different threshold levels. At the higher threshold 1.0, it boasts of high recognition rates, namely low FARs. However, FRRs are too high under side illumination. At the lower threshold 0.5, the recognition rates fall slightly and FRRs become 0% under any illumination conditions. An appropriate threshold level is determined by an application of the system.

To acquire higher recognition accuracy, PCA would need to be applied to each facial feature [9]. Alternatively, PCA could be combined with a feature-based approach [10] or neural network based methods [2].

<table>
<thead>
<tr>
<th>Illumination</th>
<th>Recognition</th>
<th>FARs</th>
<th>FRRs</th>
</tr>
</thead>
<tbody>
<tr>
<td>head-on</td>
<td>100% (15/15)</td>
<td>0% (0/15)</td>
<td>0% (0/0)</td>
</tr>
<tr>
<td>45 degrees</td>
<td>100% (14/14)</td>
<td>0% (0/14)</td>
<td>0% (0/1)</td>
</tr>
<tr>
<td>90 degrees</td>
<td>83% (5/6)</td>
<td>17% (1/6)</td>
<td>56% (3/5)</td>
</tr>
</tbody>
</table>

Table 6: System performance (threshold=1.0).

<table>
<thead>
<tr>
<th>Illumination</th>
<th>Recognition</th>
<th>FARs</th>
<th>FRRs</th>
</tr>
</thead>
<tbody>
<tr>
<td>head-on</td>
<td>100% (15/15)</td>
<td>0% (0/15)</td>
<td>0% (0/0)</td>
</tr>
<tr>
<td>45 degrees</td>
<td>93% (14/15)</td>
<td>7% (1/15)</td>
<td>0% (0/0)</td>
</tr>
<tr>
<td>90 degrees</td>
<td>77% (10/13)</td>
<td>23% (3/13)</td>
<td>0% (0/2)</td>
</tr>
</tbody>
</table>

Table 7: System performance (threshold=0.5).
4 Conclusions

We have described a system that automatically detects and recognizes a face in an image captured with a cluttered background under non-uniform illumination. The system has been tested under three different lighting conditions using 45 images from 15 persons. Our face detection scheme has achieved a fairly robust performance under various lighting conditions. Our face recognition method has achieved a significant improvement over the conventional approach. Combined with other techniques, it is possible to acquire a further improvement.

One of the features of our system is to employ a multifrequency channel strategy based on the wavelet transform. In particular, symmetry axis detection using subimages generated by the Haar wavelet can be implemented in hardware easily and efficiently. In addition, it shows a satisfactory detection performance under non-uniform illumination. This fast preprocessing helps to reduce the computational cost in the following process by narrowing down the face area.

References


Toshinki Kondo received his B.E. degree in optics and M.E. degree in information processing from Tokyo Institute of Technology, Japan in 1986 and 1988, respectively.

Since 1988, he has been employed by Canon Inc. He worked in projects related to motion detection, digital filter design, logic circuit design and stereo vision. Currently, he is working on human face detection and recognition at the University of Sydney on leave from Canon Inc. His research interests include image processing, pattern recognition, computer vision and biopsychology.

Hong Yan received his B.E. degree from Nanking Institute of Posts and Telecommunications in 1982, M.S.E. degree from the University of Michigan in 1984, and Ph.D. degree from Yale University in 1989, all in electrical engineering. From 1986 to 1989 he was a research scientist at General Network Corporation, New Haven, CT, USA, where he worked on developing a CAD system for optimizing telecommunication systems.

Since 1989 he has been with the University of Sydney where he is currently a Professor in Electrical Engineering. His research interests include medical imaging, signal and image processing, neural networks and pattern recognition. He is an author or co-author of one book, and more than 150 technical papers in these areas. Dr. Yan is a fellow of the Institution of Engineers, Australia (IEAust), a senior member of the IEEE, and a member of the SPIE, the International Neural Network Society, the Pattern Recognition Society, and the International Society for Magnetic Resonance in Medicine.
Real-Time Monitoring and Visualising the Performance of Parallel Programs

Kei-Chun Li

Department of Computing,
Macquarie University, Sydney, NSW 2109, Australia
danielli@mpce.mq.edu.au

ABSTRACT

Tools for performance measurement and visualisation have become necessary parts of programming environments for parallel computers. In many cases the users would like to monitor the performance of their applications in real-time. However, many of the existing real-time performance monitors can provide and display very limited information for performance tuning. In this paper, we introduce a monitor for supporting real-time and interactive visualisation of parallel programs performance. The monitor provides users with different levels of performance statistics, from the whole program to a single communication statement and these statistics are organised according to the hierarchical structure of the program.

1. INTRODUCTION

There are different kinds of real-time monitoring systems for different kinds of purposes. The iWarp system [2] is a real-time and interactive parallel software monitor to measure MFLOPS performance of the iWarp system. ISSOS [8] relies on the loader and operating system to provide the monitor the information, such as mapping of objects to processes and the mapping of names used in object invocations to socket identifiers used by the processes implementing an object’s operations. In the presentation of real-time monitoring, graphical visualisation of the system activity is desirable because it eases the understanding and recognition of behavioural patterns. JEWEL [4] provides views which corresponds to different performance aspects. However, the major limitation of these monitors is that they can provide very limited execution-time operations and display little information for performance tuning.

Just like post-mortem monitor, the real-time monitor is also functionally decomposed into four components, labelled: data collection, reporting, merging, and presentation. Experiments [9] show that reporting, sending measured data back to the host, was the component that most affected overhead. In post-mortem monitoring, it may be possible to store all the data locally and forward it only at the end of execution. However, in general, memory is scarce and the amount of monitoring data large, so that buffers on the slaves have to be periodically flushed to the host. To reduce this major cost, Wagner et al [9] designed an adaptive reporting scheme in system running on transputers that each node decides locally when to forward data to the host. Data are forwarded only when the sampled information (link utilisation) on the next link along the route to the host is below a predefined threshold value. However, this technique cannot be used in real-time monitoring because performance data in each processing node is needed to forward to the host immediately when the data is ready.

To provide an effective basis for interactive live visualisation of parallel programs, merging is also very important. Kimelman [3] presents an optimal technique for on-the-fly ordering and correlation of event from a number of independently produced event data streams that are being produced by a number of distinct processors in order to produce a single coherent output stream. The technique involves on-the-fly construction of the causality graph of the execution of the program. A sliding window over the graph is maintained by discarding portions of the graph as soon as they are no longer required for ensuring correct order of subsequent program events. This technique is optimal in terms of the amount of space required for the sort and the speed to produce a single coherent output stream to the rest of the system for analysis and visualisation. However, this technique cannot completely reduce the overhead of the monitoring activities.

In this paper, we introduce a monitor for supporting real-time and interactive visualisation of parallel programs performance. This monitor uses a logical clock instrumentation method which
minimises the intrusive effects of monitoring in data collection, reporting, merging, and presentation. This monitor can also appropriately direct the attention of the user by efficiently measuring the factors that are most responsible for the performance and by relating these metrics to one another and to the structure of the program. Therefore, users can tune the program in a top-down fashion, focusing their efforts on the areas that have the greatest impact on performance. Since the raw performance data in each processor is summarised as statistics during the program execution, this approach can easily provide the necessary performance information for the real-time monitoring.

The remainder of this paper is divided into five sections. The next section describes the instrumentation technique that used in the monitor to collect performance data. Section 3 discusses the design criteria of the monitor. Section 4 describes the design and implementation of the monitor. Section 5 shows the visualisation of performance data. Section 6 concludes the paper.

2. INSTRUMENTATION

Collecting performance data using software instrumentation usually suffers the intrusiveness to the original program. An intrusive monitor may alter the order of program events and distort the timing information. Parallel programs are sometimes non-deterministic in nature and hence the execution of such programs is not only dependent upon the program input, but also on the timing of the execution. Thus an attempt to monitor the runtime behaviour of a non-deterministic program through code instrumentation can potentially alter the program’s behaviour. In addition, some monitor may also hide existing deadlocks or even create new deadlock situations.

To collect accurate performance data and support the real-time monitoring of the performance of parallel programs, we have developed a logical clock approach [5] to collect the performance data in the execution of parallel programs that involves non-deterministic communications.

The basic idea behind our approach is to maintain estimates of times spent on monitoring activities and from these estimates to infer the times at which various actions would have occurred if no monitoring had been performed. The estimates form the basis for keep the execution selection and the collected timing characteristics of events the same as that in the execution of the original uninstrumented program.

The approach uses logical clocks to time and control the ordering of communication events during monitoring, and to reflect the real time execution when running without monitoring. Logical clocks are implemented by maintaining a intrusion time for each process which is the estimates of times spent on monitoring activities. Therefore, the virtual time of the occurrence of an event in a process is the real time at that moment minus the value of the intrusion time for that process. Whenever interactions occur between the monitor and the monitored processes, the intrusion time are updated.

The instrumentation is implemented as a runtime library. The library contains a set of monitor routines (or probes) which are inserted into the source code before and/or after interesting events, such as procedures, procedure calls, communication, and loops. The monitor routines are used to record execution timings and control the inter-process communication. Figure 1 illustrates the instrumentation of the communication statement. Since the monitor routines are inserted before and after the communication statement, the monitor can control the occurrences of communication events and collects accurate execution timings. To preserve the partial ordering of events, when a deterministic communication event arrives, the monitor will wait until its communication partner is ready to communicate. The monitor then calculates the time spent on synchronisation and permits the communication to start. After the execution of the communication, the monitor reads the
time and calculates the time spent for the transmission of the message. To control the occurrence of inter-process communication so as to keep the execution selection unchanged, when a non-deterministic receive event arrives, the monitor makes the decision as to which inter-process communication should happen next based on virtual times of other processes, rather than its clock time. It delays the occurrence of a communication if it is aware that there is another possible candidate process for that communication which is running in an earlier virtual time. In this way, the communication is prevented from occurring either too early in virtual time or too late in clock time. Therefore, although the real-time execution of a process is slowed down by the monitoring activities, the behaviour of the program is not changed.

Since this approach relies on communication control to preserve the partial ordering, and the virtual time hide the timing effect of monitoring activities, transparency can be achieved no matter how seriously the monitor slow down the program's execution. Therefore, this approach can be used for implementing a real-time, interactive, performance monitor.

3. DESIGN CRITERIA

Data is not the same as information. Information can help users to reduce the uncertainty. Imagine the following scenario: you are sitting in a room with a friend and you ask him "Is it raining?". You friend says "yes, it is". We could say that you have received information. But if you did not ask your friend and your friend makes the statement, "It's raining!". Could we still say that information had been passed? The answer would be "it all depends". On what? On such things as whether you knew it was raining; whether the knowledge was important to you; and the situation. Many performance tools provide information through visual forms such as space-time diagram, single event display, animation and so on. These forms do not reduce the uncertainty of users about the program's performance by a satisfactory level. Therefore, we say that they have not passed too much performance information to the users.

Imagine, now, that your friend unaccountably replies in ancient Greek. Data has been passed, but no information has been conveyed (unless you are fluent in Greek). We need another concept, i. e. a kind of protocol, to help us fully describe the situation. This scenario is analogue to our context. Many performance tuning tools provide users with volumes of complex graphs and tables that require a high degree of expertise and experience to interpret and understand. They can not convey much useful information to the tool user. We, therefore, identify the following criterion to guide our design of the performance tuning tool.

Criterion 1. The tool should present the information to help the user to minimise the uncertainty about the performance. The means of the information transmission should be direct and simple.

Effective management of information is one of the major problems faced by decision makers. A decision maker has to gather, represent, process, assimilate, and use large amount of information. In many cases, decisions have to be made in rapidly changing information environments. One way to cope with the problem of information management by decision makers is to use on-line decision-aiding tools. There is a growing consensus, however, that such on-line aiding systems become useable when they are designed to be compatible with the user's information-processing capabilities and limitations.

Designers of user-compatible on-line systems for decision processing need criteria for algorithms that search for, classify, and order the display of meaningful information chunks. The manner in which the information should be organised and presented for effective processing in interactive decision tasks depends heavily on the decision maker's cognitive strategy for chunking the information on a display. The selection of meaningful chunks is likely to be determined by contextual variables such as the perceived flow of information, the mode of presentation, e.g., textual or graphic, and the decision maker's functional objective due to differences in training, perspective, expertise, and experience.

Badre conducted a study aiming at determining the significance of chunking on the design of sequentially presented information [1]. Experiments were performed to investigate: (a) the effect on recall accuracy of the sequential display of information chunks, and (b) the effects on recall accuracy and chunking characteristics of information presented on a display which itself is viewed as a member of a sequence of displays.

The results show that chunking displayed information in a meaningful way leads to easier assimilation than if the same information is clus-
tered into an equal number of non-meaningful chunks. Also, the presentation order of meaningful chunks has an effect on ease of assimilation. If the chunks are presented sequentially in the order with which the user is familiar, then the information is assimilated faster than if the same chunks are presented in the reverse order. This is true even though the logical connectness and coherency of the information is preserved. This result implies that in the design of interactive applications where the information requires several displays, the information segmentation and presentation order should not be taken for granted. The user-information processing and management behaviour should be studied and understood. From the above discussion, we derive the second criterion.

**Criterion 2.** The performance information should be well structured and displayed to help users rapidly identity the cause and location of the performance problems.

Maddix [7] has described an experiment conducted by Klemmer and Frick in 1953 (which indicates, incidentally, that the significance of information theory was appreciated at an early date). A single white dot appeared against a black background on a 40" by 40" screen. The subjects' task was to indicate, on a grid, where the dot had appeared. The amount of information conveyed to the subject could be determined by the grid resolution. The results show that there is a constraint on the amount of information that can be transmitted in a single visual display, and it is pointless to devise more complex display than this. From this observation, we derive the third criterion.

**Criterion 3.** The performance information displayed in each window should be as simple as possible.

### 4. MONITOR DESIGN

Parallel programs incur overhead in many different ways, such as synchronisation, load imbalance, and communication. All of these aspects are important in understanding the performance of parallel programs, and that a rapid assessment of how processing time is spent in each of these aspects is extremely helpful in the performance tuning of parallel programs. To meet the criterion 1, the first goal of the monitor is to provide the performance information which includes the statistics of CPU time (for load imbalance), transmission time (for communication), and blocking time (for synchronisation), rather than the relationship of events.

The interesting events in parallel programs are procedures and communications. For long-running programs, it is assumed that most of the execution time is spent on loops, so a tool which allows the understanding of the performance at the loop level will be very useful. Figure 2 depicts the hierarchical structure of a parallel program with message passing. The top level includes the main procedure and a core which is the main body of the procedure except the monitored events. The second level includes the monitored events in the main procedure, such as communication events, procedure calls and loops. Communication events are the leaves of the tree. For a loop event, if there is any inner loop, procedure call, or communication event within the loop, a lower level description of the monitored events will be attached to the loop event. For a procedure call event, the called procedure will be attached to the calling event. To meet the criterion 2, the second goal of the monitor is to provide the user with different levels of performance statistics, ranging from the whole program to a single communication statement, which are organised according to the hierarchical structure of the program.

To achieve these two goals, our monitor uses a stopwatch approach [6] for storing and managing performance data to provide performance information rapidly and easily in real-time monitoring. The stopwatch approach records and summarises the counts and elapsed times of the interesting events in a parallel program. The execution time of each event is accumulated in dynamic records and the counts in dynamic records are incremented accordingly. All of these are processed on-the-fly during the execution of the program. There are also static tables for storing the hierarchical relationship of the events in the program. They are built before the execution of the program and can be determined from the program text. The combination of these two data structures provides performance information to the users. This approach can appropriately direct the attention of the user by efficiently measuring the factors that are most responsible for the performance and by relating these metrics to one another and to the structure of the program. Therefore, the user can tune the program in a top-down fashion, focusing effort on those areas that have the greatest impact on the performance.

Since the monitor traces all the events but stores only the statistics of the performance, it provides
Figure 2. The hierarchical structure of a parallel program with message passing

accurate performance information yet requires far less space to store trace data. To meet the criterion 3, each window of the monitor displays the performance statistics of a single particular procedure or loop structure.

Figure 3. The monitoring system on CM5

Figure 3 shows the monitor system on the Connection Machine CM5. The monitor is composed of two parts. The first part is a run-time performance monitor library. The instrumented program is replicated on all the processing nodes. During the execution of the program, the probes will collect the timing information of the events and send it to the second part of the monitor. The second part of the monitor resides on the partition manager. It collects the performance data from the processing nodes, summarises the data, and saves it to the dynamic records. After combining with the hierarchical relations stored in the static tables, it provides performance information to the users interactively.

5. DATA VISUALISATION

Figure 4 shows a sample main window of the monitor which summarises the statistics for a program being monitored. The summary breaks down the total time for the program into time
spent in computation (CPU Time), data transmission between processors (Transmission Time), and synchronisation for message passing (Blocking Time). The upper part of the window shows summarised statistics for the whole program and the breakdowns of the total time to different categories. The lower part of the window shows the breakdown of the total time spent on individual processors. The user can view the statistics from the point of view of overall program or different threads of the program running on individual processors. From the breakdowns, the user can understand the category of the performance bottleneck. When the user clicks on any of the buttons showing the time values, a tree window will be shown as illustrated in Figure 5.

The tree window displays the statistics for interesting events, such as procedure calls, message communication, or loops, depending on the category of the button pressed in the main window. All the sub-events in the monitored event are sorted according to the length of the execution time. Each event is identified by a name representing the code construct, and a line number in the original source code. For example, FOR@73 represents the for-loop in line 73. The statistics are based on the timing summarised from the category that the user chooses in the main window. For example, if the user clicks on any of the CPU time buttons in the main window, all the statistics displayed in the tree window are the times spent on computation. To view the full source code for a code object, the user can click the object’s name button to invoke the source code browser. The browser loads the appropriate file automatically, and highlights the corresponding lines of source code (see Figure 6). To further trace the performance detail of an object, the user can click the object’s time button to invoke another tree window for that object.

6. CONCLUSION

In this paper, we have introduced an interactive monitor which can real-time monitor and visualise the performance of parallel programs. The monitor uses an instrumentation technique with logical clocks to collect accurate performance data in the execution of parallel programs that involves non-deterministic communications. Since this approach relies on communication control to preserve the partial ordering, and the virtual time hide the timing effect of monitoring activities, transparency can be achieved no matter how seriously the monitor slows down the program’s execution.

The monitor provides users with different levels of performance statistics, from the whole program
to a single communication statement and these statistics are organised according the hierarchical structure of the program. This approach can appropriately direct the attention of users by efficiently measuring the factors that are most responsible for the performance and by relating these metrics to one another and to the structure of the program. Therefore, users can tune their programs in a top–down fashion, focusing their effort on the areas that have the greatest impact on performance. Since the raw performance data in each processor is summarised as statistics in its dynamic records during the program execution, this approach can easily provide the necessary performance information for the real–time monitoring. Because the data volume is not proportional to the execution time of the program, it can scale well to continuously growing numbers of processors on massively parallel computers and can handle long running programs.

REFERENCES


Model Based Hybrid Coding of Human Face Images

Mustafa Sakalli and Hong Yan

Abstract—A new hybrid model is proposed for coding of “mug-shot” face pictures as a compound of model-based coding schemes and subband, residual vector quantization (SB/RVQ). At the front edge of the model based approach, deformable templates are employed. This approach analyses the structural properties of images, to locate the expected objects, and to extract their spatial and temporal relations. Head boundaries are located by using snakes, while eye and mouth parameters are obtained by a combination of corner detectors and deformable templates. Having located the objects with their spatial interdependencies, classified codebooks are generated by utilizing a modified Laplacian pyramid structure to transform the images into subbands that are vector quantized. The same coarse and residual codebooks are used for encoding and decoding of subbands of all face images.

I. INTRODUCTION

The increasing popularity of human face recognition and coding applications such as videophone, security systems, hospitals, armies, schools and the Olympic games demands the compression for storage and bandwidth and speed that makes intelligent schemes offering model based coding (MBC) more attractive. Although the primary goal is to reduce the bit-rate for a given quality of perception, other goals are to provide a compact representation of information applicable to recognition, to provide a coding system that is independent of statistical changes of images, amenable to parallel processing and having progressive transmission ability. In this paper, we introduce a new approach to still image coding based on MBC while exploiting all means of inter-frame coding schemes (IFC). One of the most common properties of all coding systems is to provide compression by reducing the existing redundancy in any dimension and in any domain. In the case of face pictures, this occurs not just within the frame but also between image frames that share similar properties. This property of images lends itself to a system in which the compression ratio can be drastically increased for coding of similar face images [1,2] generally “mug-shot” pictures. First step is to segment the image into predefined fundamental building blocks [3], where the general definition of building blocks requires a priori information. These primary blocks in the case of face recognition, consist of eye, mouth and nose and their spatial relations. As a matching algorithm, deformable templates models these blocks as nodes of features and their spatial relations as springs. Once the face and its associated objects are located, the next challenge is to achieve very low bit rate coding ratios without being effected by anticipated blocking, ringing and quantization artifacts. Current conventional coding methods such as block DCT, KLT and waveform coding schemes, cause blocking artifacts for less than one bps, while the subband coding (SBC) diminishes this threshold until 0.3 bps. The hybrid coding scheme applied in this paper includes a modified Laplacian pyramid scheme that maintains a progressive coding structure. This provides the blocking and quantization artifacts of the lower channel to be involved within detail analysis through the high pass filtered channel of the pyramid structure. In addition to this, coarse and residual subbands are vector quantized to generate classified codebooks that are also used for encoding of other face images.

This scheme also demonstrates the predictive nature of Laplacian pyramids by using the structurally modified Laplacian pyramid [4,5,6], that yields a detail information providing less entropy to be coded. As a result of this, the adverse effects of blocking and quantization artifacts are reduced. Utilising model based structure and SB vector quantization provides the platform to generate classified codebooks of coarse and residual images. These small sized codebooks are employed for all images, so that the complexity and the computational burden introduced by the codebook design are avoided in general.

The next section of this paper introduces a comparative approach between MBC and conventional coding schemes and provides some information on deformable templates. The third section, outlines the similarities between subband structure and modified Laplacian structure. In this section also, the crucial points of vector quantization (VQ) for the proposed coding scheme are outlined to generate classified residual and coarse codebooks. Finally, implementation results are given including feature considerations.

II. DEFORMABLE TEMPLATES

Coding schemes can be classified under two major groups [7,1,2], first generation ones, where focus is given to pixel level quality and second generation coding schemes in which the message of the image is brought into the front more than pixel level quality. The purpose of second generation schemes can be expressed best with Grenander’s the pattern theory for understanding regular structures of
patterns based on the “analysis by synthesis” approach, where the purpose is to analyse the images into the elementary blocks and to synthesis back from those predetermined elementary blocks [3]. One of the handicaps of this is to define these elementary blocks, and their increasing number in a scene. This problem, in case the of face coding is simplified into eye, nose and mouth and overall face. First coding schemes, such as DCT, KLT, and Waveform coding schemes get saturated to achieve very low bit rate coding more than one bpp [1,2,8]. The others, like region and contour based and very low bit rate coding schemes, also have a limited ability to compress the image or differences of image frames more than 0.3-0.2 bpp without blocking, ringing, contour and checkerboard artifacts [8]. On the contrary, MBC achieves a high ratio of increases in compression rates especially in inter-frame coding although it has limited applications such as video phone.

Because the frames have approximately similar contents, and because of segmentation by using deformable templates, the spatial and temporal information content of the signal can be assumed as quasi-stationary [9], under same illumination conditions, i.e. details of eye, nose, mouth and face images that provides a convenient platform to use same code books for similar images. This makes the MBC an indispensable component of face coding schemes.

Figure 1: MBC employs a priori information in parameterization and in reconstruction of image components.

Unlike the first generation coding schemes, the MBC scheme changes the coarse of the coding schemes by introducing some intelligence to them [26]. The efficiency of MBC very much depends on the precise localization of objects in the scene which can be underlined as mainly a matching and image rendering problem. The classic auto correlation methods using dynamic programming suffer from computational burden where the number of computations has an exponential dependence on both template and image dimensions. As a solution to this problem, Fischler and Elschlager (1973) [10], proposed the linear embedded algorithms, in which the computational burden is decreased into a linear dependence. Fischler et al employed fixed templates, while Youlko [11] introduced flexible models of templates, deformable templates. Deformable templates approach models the objects with internal and external spatial dependencies. The internal dependencies build up an object itself, as the external one determines their interactive relations with other objects. The spatial relations between eyes and mouth can be given as a good example of this. MBC parameterizes these dependencies geometrically. Once the parameterized features of objects are determined, the following problem for MBC is to model textural dependencies. In this paper this is achieved with the inter-frame coding approach.

Figure 2: Deformable templates models the image in terms of objects (nodes) and their spatial relations (springs) between objects.

In the deformable templates approach, a network of objects is built where nodes correspond to the objects of deformable templates while the spatial relations between objects are represented by springs, figure 2. The matching between the templates and the image is provided iteratively by minimizing local fitness function $l_i(x_i)$ and global fitness function represented by spring cost functions between nodes $i$ and $k$.

$$g_{ik}(x_i, x_k) = g_{ik}(x_i - x_k),$$  

where $x$ is the positions of $i^{th}$ and $k^{th}$ features and where $i = k, g_{ik}(x_i, x_k) = l_i(x_i)$. The global fitness $G(C_p)$ and overall fitness $E(C_p)$ are given by,

$$G(C_p) = \sum_{i=1}^{p} \sum_{j=1}^{i} g_{ik}(x_i - x_k),$$  

$$E(C_p) = \sum_{i=1}^{p} \{l_i(x_i) + G(C_p)\}.$$  

respectively. The solution to minimize the total cost yields the convergence of templates into their original positions. The total cost function $E(C_p)$, equation (3), indicates the performance of total fitness. When the situation of full fitness is provided, this equation converges to zero. This
equation is a general cost of fitness that can be defined for many different geometric constraints like energy constraints of valleys, edges and peaks. Based on this, Yuille defined the first detailed [12] eye template and Lam introduced mouth-open, and mouth-closed templates [13] that are used in this paper.

Having pre-processed the face images to locate roughly their location, the parameterization of face geometry is achieved by using the snakes that was used firstly by Kass [14] for extraction of head boundaries. Once head boundaries are sketched, we use anthropologic information [15] to implement an eye and a mouth window, in which the image is scanned to seek the corresponding eye and mouth corners by corner detection algorithms. It has been proved that employing corner detection algorithms prior to deformable templates increases the convergence speed by 15-30 percent [16], thus deformable templates is used as a kind of fine tuning to obtain geometric parameters.

We introduced a model using a priori information to locate the crucial parts of the image. The following sections outline how segmented images are treated in terms of SBC and VQ which will be detailed within the next section. As has been outlined in preceding paragraphs, this kind of segmented image sequence can be assumed to be quite stationary. This property yields a platform in which the spatial and temporal dependencies are exploited by creating their classified codebooks to be used for similar kinds of images. This approach makes deformable templates a powerful tool to be employed in image localization. Besides, deformable templates can also be employed in tracking and in inter-frame coding of images to convert the signal sources into a kind of input sources with approximately constant mean and variances to be coded further down the conventional limits. The advantage of this is that a coarse template of face can be used as a rough estimation of next face images. Similar idea is especially valid for the corresponding eye, mouth and nose subimages.

III. MODIFIED LAPLACIAN PYRAMID and VECTOR QUANTIZATION

Subband Coding: Multiresolution analysis provides a natural platform for understanding of the global and the local image properties in different scales. DCT can be qualified as a SBC but the subband localization provided by DCT is rather poor [17,18]. On the other hand, SBC offers joint localization of the features in space and frequency [18]. The filter coefficients in SBC are generally calculated for an optimum solution to minimize the resulting aliasing due to sampling procedures and to obtain an approximate or perfect reconstruction (PR) of the original signal back in the absence of any quantization artifacts. Many different types of subband filter structures are possible to split the original signal band from dyadic (equal bandwidth filter banks) to unequal band-width multirate filterbanks (biorthogonal filter banks) [6,17]. The crucial point in the application of SBC approach is the selection of filter coefficients where employing ideal filters, “brickwall”, produces undesirable ringing effects in the spatial domain. The filter banks with overlaying spectrums of each other can be employed so that two important conditions, no distortion (unity) and anti-aliasing conditions can be satisfied to provide perfect reconstruction of the original signal back.

![Figure 3: Application of modified Laplacian pyramid for segmented face images.](image)

In this paper we applied a modification of Laplacian Pyramid [6]. The Laplacian pyramid structure which is originally proposed by Burt and Adelson [4], actually has a dyadic-like three structure. Figure 3 is drawn for 1D case where the input signal $x_i(n)$ of $i_{th}$ stage is blurred by a gaussian filtering followed by decimation $v_i(n)$ and then convolution with an interpolation filter of synthesis channel, $x_i(n)$. At each stage of pyramid resolution, two outputs are obtained, one blurred $v_i(n)$ with size $\frac{n}{2}$ and detail image $x_i$ with some size of original signal, where subscribe $r$ refers to the residual. The residual signal $x_{ir}$ is obtained as the difference between, interpolated $x_{iL}(n)$ and original in each $i_{th}$ stage of analysis and synthesis,

$$x_{ir}(n) = x(n) - x_{iL}(n)$$

The pdf of residual image $x_{ir}(n)$ is Laplacian for the most stationary and gaussian signal sources. In each step of filtering, the bandwidth of the input signal is divided in a dyadic fashion. An obvious drawback of Laplacian pyramid structure is the expansion of information into one and half times the original input size at each stage. This can be prevented with the provided modification in other words by further filtering and down-sampling of residual.
signal [9][8], as it is performed separately through the higher frequency channels of SBC schemes for the original image, figure 2. The caution should be taken in determination of filter coefficients that should satisfy the conditions of no distortion and anti-aliasing. In the case of no quantization and coding artifacts, the anti-aliasing conditions are satisfied with selection of interpolation filter coefficients as, \( G_0(z) = -z^{-1} H_1(-z) \) and \( G_1(z) = z^{-1} H_0(-z) \). This yields no distortion condition. Two possible approaches to select analysis filter coefficients can be given for the perfect reconstruction condition to be satisfied. One approach, for equal bandwidth splitting, with equal length high-pass and low-pass filters, employs paraunitary pairs of \( H_0(z), H_1(z) \), where

\[
H_1(z) = z^{-(N-1)} H_0(-z^{-1})
\]

and the amplitude responses are mirror images [6] of each other,

\[
H_1(e^{j\omega}) = H_0(e^{j(\omega-\pi)})
\]

(5)

For the 2-channel PR-QMF, the paraunitary solution implies the time-domain orthogonality conditions,

\[
\sum_k h_i(k) h_i(k + 2n) = \delta(n), i = 0, 1
\]

\[
\sum_k h_i(k) h_0(k + 2n) = 0
\]

(6)

The second alternative is to select image coefficients satisfying anti-aliasing condition which is proposed by Vetterli and Herley, 1992, [5][6], by using biorthogonal filters with conditions,

\[
\sum_k h_0(k) \hat{g}_0(k - 2n) = \delta_n \sum_k h_1(k) \hat{g}_0(k - 2n) = 0
\]

\[
\sum_k h_1(k) \hat{g}_1(k - 2n) = \delta_n \sum_k h_0(k) \hat{g}_1(k - 2n) = 0
\]

(7)

where \( \hat{g}_i(n) \delta = g_i(-n) \). This biorthogonal approach provides unequally split band-widths of filters, with unequal length. Details of derivations can be obtained in [6]. The potential points of a modified Laplacian pyramid structure especially for image coding is emphasized within the next implementation section of this paper. The subbands of coarse and detail images are vector quantized. The detail subbands are obtained just for crucial parts of image, such as eye, nose and mouth to be processed further.

**Vector Quantization:** A variety of quantization methods can be brought into consideration [19], among those, scalar quantization is the simplest to implement while VQ offers better compression ratios for relatively increased complexity. VQ is an appealing coding technique because the rate-distortion bound can be approached by increasing vector dimension. However, its complexity and computational burden and its storage requirements expands exponentially with increasing dimension \( k \) of vectors. This problem is diluted by using normalized stationary or quasi-stationary gaussian and small sized signal sources and by building classified codebooks for fundamental specific parts of the face. For a number of reasons stemming from the memory and computationally efficient structure of the RVQ, it has been shown that RVQ usually achieves compression ratios competitive with those of JPEG and the others. [20][21].

![Figure 4: The overall system proposed as a combination of MBC and SBC for face database coding.](image)

Vector quantization (VQ) can be defined as a mapping from \( k \)-dimensional vector space of signal source with different distribution of pdf into another space of ordered finite sets \( \mathbb{R}^k \) that are defined as reproduction codewords or codewords, satisfying the required constraints of minimum distortion with limited \( M \) number of codevec-
tors $M < N$. Assume that we have a set of stationary
$N$ vector sources $S_i$, to be coded under the same con-
tions of performance criteria given by minimizing overall
distortion measure and limited number of codebooks for
$i \in N = \{1..N\}$. Then, the object of VQ is to form a fam-
ily $C = \{v_j\}$ for $j \in M = \{1..M\}$ of the codevectors, that
is called codebook, with minimum overall distortion to re-
construct the source. Both sides of VQ, vector encoding
and decoding, employ the same code book. The encoder
assigns an index $c$, $c \in 1..j$ of codeword that yields the
minimum distance to each input source $s_i$ as the decoder
converts the index $c$ to the corresponding codeword $v_c$ in
the codebook $C$. The Euclidean distortion measurement
between each vector source and its decoded corre-
respondence is given by

$$Ed(s_i, v_c) = ||s_i - v_c||^2 = \sum_{j=0}^{k-1} (s_{ij} - v_{cj})^2$$  \hspace{1cm} (8)

By choosing the number of codevectors and the size of
codebook, the bit-rate per symbol of codewords and the
compression rate is controlled. The higher the number
of codewords and the smaller the dimension, the higher
the signal to noise ratio and the lower the compression
will be maintained. The key issue in VQ is to obtain
an optimum code book for each image or for a class of
gaussian stationary sources. There are several techniques
for designing an initial codebook and obtaining an opti-
mum codebook. Generalized Lloyd clustering algorithm,
proposed by Linde, Buzo and Gray calculates the cen-
troids of evenly spaced vectors to converge to an initial
codebook [24]. Then, it iteratively minimizes distortion
between the input vectors and corresponding codewords.
The centroids of evenly spaced $k$ dimensional $T$ source
vectors is calculated as an initial codebook,

$$C_{i=0}^k = \frac{1}{T} \sum_{i=1}^{T} s_i^T$$ \hspace{1cm} (9)

where $s_i^j$ is the $j^{th}$ component of the $i^{th}$ source vector for
$i \in \{1..k\}$ and $T$ initially is equal to $N$ but in the follow-
ing stages it equals the number of source $s_i$ vectors allo-
cated to each centroid. We used a splitting technique to
increase the number of codewords to the desired number
$M$ by perturbing for splitting to new possible codevectors
[24]. In this study, the stochastic relaxation is employed
to control the amplitude of perturbation [19], which is re-
duced with each attempt of convergence in parallel to the
variance of allocated vectors to this cluster of the centroid.
Following splitting, the training vector source is split of
into the number of split codebooks. Each source vector $s_i$
is allocated to the closest codeword $v_c$ in Euclidean space
for mapping,

$$\min \{ Ed(s_i, v_c) \}_{c=0}^T \implies s_i \in v_c$$ \hspace{1cm} (10)

The next step is to check if this new mapping can give
rise a new decoder by using previous equation (9) to get
new centroids of clustered vectors allocated to old cen-
troids. This is a kind of fine tuning towards a possibly
closer positioning of the local centroids, by calculating
overall distortion for new location of centroids. The over-
all distortion can be driven from equation (8) by taking
the summation for each source vector between source vec-
tor $s_i$ and their new centroids $v_c$ of clusters. These steps
are iteratively repeated until the desired number $M$ of
codewords is arrived at. Once the predetermined number
of codewords are obtained, iteration continues to find a
better possible local minima within the close vicinity of
the suggested codewords set in each attempt until overall
distortion starts to diverge.

Codebook generation for each image is an expensive
process. We introduced classified codebooks to be used
for normalized images by the assumption of stationary im-
age sources captured by deformable templates instead of
reproducing them for each image again and again. That
increases the efficiency of this coding scheme. The over-
all implementation considerations for the deformable tem-
plates and composite structure of the SBC and RVQ are
given in the next chapter.

Figure 5: (a) Original image, and (b) head bounded by
snakes, (c) Normalized so that the distance between two
cyc centers is 32 pixel.

IV. OVERALL IMPLEMENTATION

An MBC can be achieved in five stages including the three
stages of classic compression technique: transformation of
the information, quantization and entropy coding [17]. All
MBC schemes require a priori information of the expected
shapes for the objects in the scene including deformable
templates, figure 1. The active contour model represented
by snakes is used in face boundary extraction. Snakes
is an energy-minimizing spline guided by external forces
and image forces, that draw it towards the boundary of
the feature. These methods improve the reliability of ex-
tracting global features [13][25], however, both methods
require initial estimations based on a priori information
[10,14]. These initial estimates affect the performance of
contour representation and the time required to locate the
features.
The extraction of the face boundaries is performed as a first step by using snakes [13-14]. Then the approximate location of the other facial features are estimated based on anthropometric measures [15]. The contour is initially placed near the image under consideration, then image forces draw the contour to the edges of the region under consideration. A snake can be represented as a parametric curve \( v(s) = (x(s), y(s)) \), where the parameter \( s \) refers to the arc length. An energy functional of the snake is defined as

\[
E_{\text{snake}} = \int_0^1 E_{\text{snake}}(v(s)) ds
\]

\[
= \int_0^1 E_{\text{internal}}(v(s)) ds + E_{\text{image}}(v(s)) + E_{\text{constraint}}(v(s)) ds
\]

(11)

where \( E_{\text{image}} \) and \( E_{\text{constraint}} \) are the image and the external constraint forces, respectively and \( E_{\text{internal}} \) is the internal energy of the contour due to bending and discontinuities. In this work we implemented greedy snakes [16,27,28] to minimize energy functional of the contours,

\[
E = \int [\alpha(s) E_{\text{continuity}} + \beta(s) E_{\text{curvature}} + \gamma(s) E_{\text{image}}] ds
\]

(12)

where \( E_{\text{continuity}}, E_{\text{curvature}} \) and \( E_{\text{image}} \) are the continuity, the curvature and the image forces weighted with \( \alpha, \beta \) and \( \gamma \), respectively. Figure 5a and 5b illustrate the original faces with the 112x92 pixel size taken from Olivetti face database and its contour bounded by using snakes. The frame of face images are painted to an average pixel value for all images determined as 130 for the efficiency of the codebooks.

After locating the face boundaries, the localization of eyes and mouth are performed within the windows whose dimensions are determined according to the spatial anthropometric relations. We employed corner detection algorithms [13, 27, 28] to scan eye and mouth within these windows, firstly by searching noncollinear lines under orientation constraints by using morphological edge detectors [27], than the eye and mouth corner candidates are determined by using the valley detectors [25] and region dissimilarities [28]. A cost function is defined based on the relative distances between two corners, and the region dissimilarities which is formulated as the difference between average grey level intensities of regions. The next stage is the initialization and extraction of the eye and mouth features performed by eye templates [11,12] and mouth templates [13]. Figure 6a shows segmented eye and mouth images for a normalized face. All images are normalized to 32 pixel distance between two eye centers figure 5c that are determined after the convergence process of templates are being completed. This normalization keeps relative spatial dependence of features unchanged and for the sake of simplicity the all image size are taken with the size of 80 X 64 as fold of four for two level decimation and interpolation when the background around the face is discarded.

Figure 6: (a) Normalized image (by using deformable templates) is segmented into sub components, (b) illustrates two level wavelet transform of normalized image, seven subbands split in a dyadic fashion from top to left: LL={LLLL, LLLH, LLHL, LLHH}, LH, HL, HH, (c) and the inverse WT of the same image by using just four sub-sections of subband LL, after subbands are vector quantized. Blocking artifacts are visible. Daubecies 8 tap binominal filters are used.

Figure 7: These images are reconstructed after subbands are vector quantized and subbands are obtained by using modified Laplacian pyramidal structure. The first image (a) corresponds to the reconstruction from the lowest band LLLL, (compressed by 0.0115 bpp), the next image (compressed by 0.05 bpp) and reconstructed after the residual is decomposed and reconstructed by adding three vector quantized subbands of LLLL, LLHL, LLHH of residuals, and (c) the last image (compressed to 0.065 bpp) is assembled with the contribution of the lowest LLLL subbands of eye, nose, and mouth residuals in addition to (b). Over all compression ratio achieved is found to be 130 with respect to the normalized image.

The composite diagram of coding scheme including all stages of MDC and SB RVC is given in figure 4. SBC scheme employed for modified Laplacian pyramid uses an eight tap PR-QMF binominal filters of Daubecies wavelets [29]. Initially the two layers of analysis stages are applied to get the lowest band with the size of one sixteenth of
original image size. Circular convolution is performed to diminish the ringing effects towards the edges of the image. All segmented image dimensions are chosen larger than filter dimensions. The codebook obtained from this subband is used as a codebook for all incoming face images. The codevector number of eight is selected with 4 by 4 dimension. In addition to this the other size of codevectors are also experimented. Vector quantization is the crucial stage where compression is provided. The capacity of the codevector is exploited in each case by normalizing the signal that will be coded by using the same codebooks. The residual image of all the face is obtained, as a difference between the original and the convolved one with the interpolation filters. Theoretically this image corresponds to the summation of all higher channel subbands and blocking errors that are introduced from block transformation of the image and the spillover from one channel to the others due to quantization artifacts. Further two layer analysis is performed for the residual image that involves the analysis through the high frequency channels and one more low frequency channel analysis. At the second layer of the modified Laplacian pyramid, the lowest three bands of residual image, LLLL, LL LH, LLLH are vector quantized. This leads to the allocation of 240 bits where 60 bits are sacrificed for each subband of low frequency range and middle frequency range bands of residual image. In the entropy coding stage, fixed length coding is preferred due to its simplicity and there is no much gain expected from variable length coding. Based on the parameterized models, the residual image sizes for eye, nose and mouth windows are to be selected 24x24, 24x24 and 16x16 pixel height and width, respectively. One level of SB analysis is performed for eye residual images that yields subband size of 12 to 12 and the lowest band of this is vector quantized with 2 to 2 dimensional four codewords of the eye codebook that results in 9x2 bits allocation for each eye. Similarly with the addition of bits required for subband and vector quantized mouth and nose residual images, we have achieved to reconstruct a recognizable face image close to its original within length of 320 bits with entropy coding. Figure 7 demonstrates the reconstructed face images from first interpolation, \( x_1[n] \) in figure 7a, then, second interpolation \( x_2(n) \), figure 7b is reconstructed by the addition of three vector quantized subbands LLLL, LL LH, LLLH of residual image \( x_1[n] \) and \( x_1(n) \). The last image in figure 7c is assembled by combining two level analysis and interpolation of residual eye, nose, mouth images with the image in figure 7b. This corresponds to a compression ratio of 130 comparing to framed image of figure 5c and approximately 1/250 times of the original image figure 5a. Figure 8 demonstrates the original bounded face, reconstructed face when overall procedures are repeated again for different convergence of codebooks and the error image that is scaled for visual purposes. Experiments are performed on the Sun SPARC 2 workstation. The procedures of the subband analysis and synthesis are repeated for all images, but generating codebooks is performed just once. The same codebooks are used for all face images to reduce the computational burden of VQ.

Figure 8: (a) Original normalized image, (b) reassembled image that is synthesised by the vector quantized lowest band of coarse image and the other three vector quantized subbands of residual face image and the lowest bands of eye, nose and mouth residuals. (c) Scaled difference image between reconstructed (b) and original (a).

IV. CONCLUSION and FUTURE CONSIDERATIONS

In this paper we have demonstrated a new approach to face database coding that is applicable to the coding of other kinds of images by providing very high compression ratios: approximately 40 bytes for a face with the dimensions of 80x64 pixel. We have implemented the context based classified residual codebooks of objects. The computational burden of vector quantization on that scheme is rather shifted towards the localization of images and the determination of required parameters by using MBC approach. It is possible to increase the speed of vector quantization by using hierarchical approaches [20,33] and possible to decrease the codebook sizes further down by exploiting interdependencies between codebooks. The next step of this study is to apply biorthogonal filters [17, 5] to increase the quality of the reconstructed image.

REFERENCES

[9] A. Papoulis, Probability, Random Variables and Stochastic Pro-
from Faces Using Deformable Templates, Int. J. Comput. Vision,
[16] K.M. Lam and H. Yan, Fast Algorithm for Locating Head
1994.
[17] M. Vetterli and J. Kovacevic, Wavelets and Subband Coding,
Entropy-Constrained Residual VQ with Application to Image Cod-
[22] B. Ramamurthi and A. Gersho, Classified Vector Quantization
1115, Nov. 1986.
1980.
[26] H. Li, A. Lundmark and R. Forchheimer, Image Sequence Cod-
ing at Very Low Bit Rates: A review, IEEE Trans. Image Proc.,
[27] D.J. Williams and M. Shah, A Fast Algorithm for Active Contour
and Curvature Estimation, CVGIP: Image Understanding Vol 55,
[28] K.-M. Lam and H. Yan, Locating and Extracting the Eye in
Human Face Images, J. Pattern Recog. Soc., Vol. 9, No. 5, pp
771- 779 May 1996.
Transform for Multiresolution Signal Decomposition IEEE Trans.

Mustafa Sakalli was born in Denizli, Turkey. He received the B.E. degree in Electrical Engineering from the Technical University of Istanbul, Turkey, in 1980 and the M.S. degree from the Institute of Biomedical Engineering of Bogazici University, Istanbul. He is currently engaged with his Ph.D research with the Imaging Group at the Department of Electrical Engineering, University of Sydney.
Fixation Point Estimation and Transfer

Miroslav Trajković and Mark Hedley
Department of Electrical Engineering,
University of Sydney, NSW 2006, Australia
email: [miroslav, hedley]@ee.usyd.edu.au

Abstract—Fixation of the gaze on one point in the environment has been recently proposed as a stable approach to tracking and structure from motion estimation. A problem which has been addressed by many researchers is how to estimate and track fixation point with highest possible stability.

In this paper, we propose a statistically based approach to fixation point estimation and transfer. The method assumes affine camera model and makes use of corner detector and matcher over three consecutive frames. The fixation point has been estimated for each cluster of corners using the affine structure available from three views. It has been represented as a linear combination of all cluster points and corresponding weights were found using Minimum Variance Estimator. Transfer has been performed over three views using similar principles and was shown to be insensitive to individual corner dropout or reappearance.

I. INTRODUCTION

Given a sequence of images taken by moving observer the aim of our research is to separate between background motion (caused by motion of the camera) and independent motion (caused by independently moving objects in the scene) and to track independently moving objects over the time. The range of applications is fairly wide varying from traffic control and autonomous mobile robots to the multimedia and video coding. To achieve a given goal we divided our task in several steps:

- **Velocity measurement**, i.e. computation of velocities of the points of the scene.
- **Motion Segmentation, or Clustering**, i.e. grouping points with similar velocities into objects.
- **Tracking**, track segmented objects over the time and provide feedback to improve the segmentation.

In general velocities in an image sequence can be estimated through: 1) Optical flow computation; or 2) Feature matching. The former usually gives a dense velocity field i.e. velocities are computed at every pixel location (although the reliability of the velocity estimate varies), while the latter usually gives velocities of some distinctive features of the scene such as corners, edges, lines, curves, regions etc. It is the latter approach which is employed in our work and this was selected for two reasons. First, feature matching algorithms are in general more robust and better handle occlusion and motion discontinuities; and second, the number of velocity vectors computed by feature matching methods is much smaller, typically by two orders of magnitude. This greatly reduces the computational effort and makes this approach more attractive for real time applications such as motion segmentation, tracking or shape finding.

The features that we detect are corners and the advantages of using them are explained elsewhere [1]. To find corners we used either a modified version of the Harris corner detector [2] or the corner detector that we have developed [3, 4].

The corners are matched using a correlation based algorithm [4]. For each corner $C_1$ in the first image we find the most similar corner within a region of interest (usually a rectangle centred on $C_1$) in the second image using a correlation between small patches around every feature as a measure of similarity.

This procedure is then repeated in the backward direction. For each corner in the second image we find the best match in the first image. Only mutually best candidates are accepted, while all the others are disregarded.

Unfortunately this procedure still leaves a significant number of mismatches. In order to reduce this, a temporary constraint is employed and three images are taken for matching. Features from the second and third images are matched and only those with matches in all three images are taken into account. This simple procedure decreases the number of strong matches, but, more importantly, it dramatically reduces the number of mismatches. It is not however recommended to use the same procedure over longer sequence of images, because as the length of the sequence increases all the corners will have mismatches and no corners will left.

Motion segmentation is done by clustering, i.e. grouping together points with similar motion. To do this, we have developed two clustering algorithms [5] - robust (for smoothing) and gravitational clustering. Both algorithms work for an unknown number of clusters and can be implemented to higher dimensional feature space. Other clustering methods could be implemented as well, e.g. [6].

Tracking is an extremely important part in any motion analysing system because it makes temporal connection between measurements and gives the opportunity of error
correction. As mentioned before, while an individual corner may provide a stable track for few frames, either noise or occlusion will inevitably cause it to disappear sooner or later, making a native tracking (i.e., only matching) impossible. A good solution to this problem is "fixation" over time [7, 8]. Instead of tracking all the points over the time we track only one virtual point, which is called fixation point, and we track it reliably using method called transfer. Then, through the closed loop system, we can recover positions of all the features, including their positions in frames where they are "missing".

To perform affine transfer over long sequence of images it is necessary to have at least four matched points (usually called basis points) over each set of three consecutive images, and the choice of basis points is of big importance here. It has been shown [9] that this choice affects the stability of transfer and several heuristic methods for choosing stable basis set have been proposed, but little work (except [7, 10]) has been done on using all available points for accurate transfer. In this paper, we propose statistically based method for optimal estimation of fixation point using all available points over first three frames. Transfer is also performed in the optimal way, using all available points.

The paper is organised as follows. In Section II we introduce affine camera model and give necessary background for the affine transfer algorithm. In Section III we explain in details a novel affine transfer method, and in Section IV we give conclusion remarks and some directions for future work.

II. BACKGROUND

To perform affine transfer we assume an affine or weak perspective camera projection. This model is valid when change in depth is small comparing to average distance from the optical center, and this condition is almost always satisfied for independently moving objects. Mathematically, affine camera can be represented by equation:

\[ \mathbf{x} = \mathbf{Mx} + \mathbf{t} \]  

(1)

where \( \mathbf{x} \) is a \( 2 \times 1 \) image position vector, \( \mathbf{M} \) is a \( 2 \times 3 \) matrix, \( \mathbf{X} \) is a \( 3 \times 1 \) world position vector, and \( \mathbf{t} \) is a \( 2 \times 1 \) translation vector.

Consider now set of four points, \( \mathbf{O}, \mathbf{A}, \mathbf{B} \) and \( \mathbf{C} \), in general position (i.e., non-coplanar) on an object under surveillance. The four points define a basis set, say \{\( \mathbf{A} - \mathbf{O}, \mathbf{B} - \mathbf{O}, \mathbf{C} - \mathbf{O} \)\} and relative to them, each point \( \mathbf{X} \) may be uniquely defined by three affine coordinates \( \alpha, \beta \) and \( \gamma \) as:

\[ \mathbf{X} = \alpha(\mathbf{A} - \mathbf{O}) + \beta(\mathbf{B} - \mathbf{O}) + \gamma(\mathbf{C} - \mathbf{O}) + \mathbf{O} \]  

(2)

These coordinates are invariant to the affine projection in the sense that corresponding affine coordinates are the same linear combination of the projected basis vectors, i.e.:

\[ \mathbf{x} = \mathbf{MX} + \mathbf{t} = \alpha(\mathbf{a} - \mathbf{O}) + \beta(\mathbf{b} - \mathbf{O}) + \gamma(\mathbf{c} - \mathbf{O}) + \mathbf{o}. \]  

(3)

Given two views of four basis points, we can compute affine coordinates of the fifth point \( \mathbf{X} \) in two views by solving overconstrained system of (two vector or four scalar) equations:

\[
\begin{bmatrix}
\mathbf{x} - \mathbf{o} \\
\mathbf{x}' - \mathbf{o}'
\end{bmatrix} =
\begin{bmatrix}
\mathbf{a} - \mathbf{o} & \mathbf{b} - \mathbf{o} & \mathbf{c} - \mathbf{o} \\
\mathbf{a}' - \mathbf{o}' & \mathbf{b}' - \mathbf{o}' & \mathbf{c}' - \mathbf{o}'
\end{bmatrix}
\begin{bmatrix}
\alpha \\
\beta \\
\gamma
\end{bmatrix}.
\]  

(4)

After computing affine coordinates of a point \( \mathbf{X} \), and having the projected positions of reference (basis) points in a novel view it is straightforward to determine projection of \( \mathbf{X} \) in the novel view as:

\[ \mathbf{x}' = \alpha(\mathbf{a}' - \mathbf{o}') + \beta(\mathbf{b}' - \mathbf{o}') + \gamma(\mathbf{c}' - \mathbf{o}') + \mathbf{o}'. \]  

(5)

The equation (4) and (5) are very important since they allow us to perform tracking of an object over long sequence of images, providing that in each three consecutive frames there are at least corner correspondences, a condition that is easily satisfied in practice. The tracking is performed on the following way:

1) Define fixation point \( \mathbf{G} \) in the first two frames \( \mathbf{g}, \mathbf{g}' \).

2) Using formula (4) compute affine coordinates \( (\alpha_\gamma, \beta_\gamma, \gamma_\gamma) \) of the fixation point.

3) Having affine coordinates, find the position \( \mathbf{g}'' \) of fixation point in the third frame.

4) The process is then repeated over frames two, three and four (step 1 is omitted because position of fixation point has been transferred) and so on.

Several issues however need to be addressed further:

1) The choice of basis points. We are often given more than four points; how to choose basis points to do the transfer? On the other hand, why not to use all the available points, and how?

2) Given a set of basis points, how to optimally chose fixation point.

In this paper, we propose statistically based method for optimal estimation of fixation point using all available points over first three frames. The method is then extended to affine transfer, which is also performed in the optimal way, using all available points.

III. AFFINE TRANSFER ALGORITHM
Let us suppose that we have matched points \( x_1, \ldots, x_n \) over three frames with corresponding matches being \( x'_1, \ldots, x'_n \) and \( x''_1, \ldots, x''_n \). Then, it holds:

\[
x_i = M X_i + t,
\]

\[
x'_i = M' X_i + t',
\]

\[
x''_i = M'' X_i + t'';
\quad i = 1, \ldots, n.
\]

or, in compact form

\[
W = \begin{bmatrix} M \\ M' \\ M'' \end{bmatrix} L + \begin{bmatrix} t \\ t' \\ t'' \end{bmatrix},
\]

where

\[
W = \begin{bmatrix} x_1 & x_2 & \cdots & x_n \\ x'_1 & x'_2 & \cdots & x'_n \\ x''_1 & x''_2 & \cdots & x''_n \end{bmatrix}
\]

is a measurement matrix and

\[
L = \begin{bmatrix} X_1^T \\ X_2^T \\ \vdots \\ X_n^T \end{bmatrix}^T
\]

is the shape vector. System (6) is overdetermined and to solve for \( M, M', M'', t, t', t'' \) and \( L \) we have to employ least squares technique. Note that \( M \)'s and \( L \) can not be uniquely solved, but as it will be seen later, it is not necessary. The cost function to be minimised by LS is obviously:

\[
e(M, M', t, t', x_i) = \sum_{i=1}^{n} |x_i - M X_i|^2 + \sum_{i=1}^{n} |x'_i - M' X_i|^2 + \sum_{i=1}^{n} |x''_i - M'' X_i|^2
\]

By minimising (7) directly over \( t, t' \) and \( t'' \) we get:

\[
t = \bar{x} - M \bar{X},
\]

\[
t' = \bar{x}' - M' \bar{X}
\]

\[
t'' = \bar{x}'' - M'' \bar{X},
\]

where \( \bar{x}, \bar{x}', \bar{x}'' \) and \( \bar{X} \) are the centroids of data sets \{\( x_i \}\}, \{\( x'_i \}\}, \{\( x''_i \}\} and \{\( X_i \}\}, respectively. By substituting these offsets in above equation (6) and writing \( \Delta x = x_i - \bar{x} \) etc., we obtain:

\[
\hat{W} = \begin{bmatrix} M \\ M' \\ M'' \end{bmatrix} \Delta L
\]

where

\[
\hat{W} = \begin{bmatrix} \Delta x_1 & \Delta x_2 & \cdots & \Delta x_n \\ \Delta x'_1 & \Delta x'_2 & \cdots & \Delta x'_n \\ \Delta x''_1 & \Delta x''_2 & \cdots & \Delta x''_n \end{bmatrix}
\]

is called registered measurement matrix. Note that \( \hat{W} \) is a product of \( 6 \times 3 \) and \( 3 \times n \) matrices and hence it’s rank is at most 3.

The SVD of \( \hat{W} \) may be written as

\[
\hat{W} = U S V^T,
\]

and the optimal estimate of \( \hat{W} \) is given as

\[
W_{opt} = U_3 S_3 V_3
\]

where index 3 means that only columns or rows corresponding to the three largest singular values are taken into account, since other singular values correspond to noise.

We now have measurements \( \hat{W} \) and predicted values \( W_{opt} \). The closer a predicted value is to the measurement the more confident measurement is, and we can define variance of each point \( (x_i, x'_i, x''_i) \) as a function of difference between measured and predicted values. For example, let

\[
\Delta W = \hat{W} - W_{opt} = \begin{bmatrix} \Delta w_1 & \Delta w_2 & \cdots & \Delta w_n \\ \Delta w'_1 & \Delta w'_2 & \cdots & \Delta w'_n \\ \Delta w''_1 & \Delta w''_2 & \cdots & \Delta w''_n \end{bmatrix}
\]

then variance \( \sigma_i \) could be defined as

\[
\sigma_i^2 = \Delta w_i \Delta w_i T + \Delta w'_i \Delta w'_i T + \Delta w''_i \Delta w''_i T.
\]

**The Initial Estimation of Fixation Point**

The fixpoint is defined as a linear combination of all available points:

\[
\mathbf{G} = \sum_{i=1}^{n} \alpha_i X_i,
\]

and corresponding point in image plane is

\[
\mathbf{g} = M \mathbf{G} + t = M \sum_{i=1}^{n} \alpha_i X_i + t
\]

\[
= \sum_{i=1}^{n} \alpha_i (M X_i + t) \left( 1 - \sum_{i=1}^{n} \alpha_i \right)
\]

\[
= \sum_{i=1}^{n} \alpha_i x_i + t \left( 1 - \sum_{i=1}^{n} \alpha_i \right)
\]

In order to preserve “invariantness” to affine projection, it has to be

\[
\sum_{i=1}^{n} \alpha_i = 1, \quad (9)
\]

and then

\[
\mathbf{g} = \sum_{i=1}^{n} \alpha_i x_i
\]

Since matches \( x_i \) are random variables, \( \mathbf{g} \) is random variable as well and its variance is
\[ \sigma_g^2 = \sum_{i=1}^{n} \alpha_i^2 \sigma_i^2. \] (10)

The fixation point will be the most stable for those values of \( \alpha_i \) which minimise variance \( \sigma_g^2 \) and we find this values by minimising (10) under constraint (9). This is easy to solve and corresponding values are:

\[ \alpha_i = \frac{1}{\sum_{i=1}^{n} \frac{1}{\sigma_i^2}}. \] (11)

**Subsequent Fixation Point Estimation**

Given a set of matches over next three consecutive views (second, third and fourth \( y_1, y'_1, y''_1, \ldots, y_m, y'_m \) and \( y''_m \) and position of fixation point in the frames two and three \( g \) and \( g' \) (upper scripts are changed for clarity) our goal is to estimate fixation point in the fourth view.

Again, we write fixation point as a linear combination of the positions of the matches

\[ g = \sum_{i=1}^{n} \beta_i y_i, \] (12)

and the variance of \( g \) is again

\[ \sigma_g^2 = \sum_{i=1}^{n} \beta_i^2 \sigma_i^2. \] (13)

and \( \sigma_i \) are computed using previously described procedure.

Coefficients \( \beta_i \) are determined so that variance (13) is minimised and that following constraints are satisfied:

\[ \sum_{i=1}^{n} \beta_i = 1 \]
\[ \sum_{i=1}^{n} \beta_i y_i = g \]
\[ \sum_{i=1}^{n} \beta_i y'_i = g'. \]

This is solved by using Lagrange multipliers and it can be shown that

\[ \beta_i = \frac{\mu_1 + \mu_2 y_i + \mu_3 y'_i}{\sigma_i^2} \]

where \( \mu_1, \mu_2 \) and \( \mu_3 \) are solutions of the system:

\[ \mu_1 \sum_{i=1}^{n} \frac{y_i}{\sigma_i^2} + \mu_2 \sum_{i=1}^{n} \frac{y'_i}{\sigma_i^2} + \mu_3 \sum_{i=1}^{n} \frac{y''_i}{\sigma_i^2} = 1 \]
\[ \mu_1 \sum_{i=1}^{n} \frac{y_i}{\sigma_i^2} + \mu_2 \sum_{i=1}^{n} \frac{y'_i}{\sigma_i^2} + \mu_3 \sum_{i=1}^{n} \frac{y''_i}{\sigma_i^2} = g \]

Note that the above system has five unknowns and that \( \mu_1 \) and \( \mu_2 \) are vectors.

The fixation point in the last (fourth) frame \( g'' \) is simply computed as

\[ g'' = \sum_{i=1}^{n} \beta_i y''_i \]

**IV. CONCLUSION**

In this paper we presented an algorithm for fixation point estimation and transfer over long sequence of images. Each point in the set of matches is represented as a gaussian random variable with the variance proportional to the distance between measured and position predicted by the best fitting affine motion model. The fixation point is then estimated as a linear combination of all matches using Minimum Variance Estimator. Transfer is performed in a similar manner, although more constraints have to be satisfied.

At the moment we are investigating algorithms for the outlier detection and are implementing our algorithm on real and synthetic realistic sequences. We are also developing an algorithm for sequential estimation of motion and structure under affine motion paradigm.

**REFERENCES**


Miroslav Trajković was born in Bujanovac, Yugoslavia. He graduated from the faculty of Electrical Engineering, University of Niš.
achieving the shortest study time in the history of the University. As an undergraduate student he competed at Yugoslav wide Competitions of Electrical Engineering students in Mathematics and Telecommunications, winning first and second place respectively. He was awarded several "student of the year awards" and "the best graduated student award". His Diploma Work "Inequalities in interval mathematics" was pronounced the best Diploma Work in 1993. In 1993, he joined Numerical Mathematics Group at the Faculty of Electrical Engineering, Univ. of Niš, working on Interval Mathematics, and published several papers in respected international journals.

In 1995 he has joined the SUEE where he is now working on motion estimation under supervision of Dr Mark Hedley. His research interest include Computer Vision, Motion Estimation and Robust Statistics. He is member of IEEE, British Machine Vision Association and Australian Pattern recognition Society.
Rotational Motion Artifact Correction in MRI
Chaminda Weerasinghe and Hong Yan

Department of Electrical Engineering
University of Sydney
NSW 2006, Australia.
E-mail: chaminda@ee.usyd.edu.au

Abstract—Rotational motion artifacts are prevalent in magnetic resonance (MR) images of head and brain. Even a slight rotation during the data acquisition can have a significantly adverse effect on the diagnostic value of the image. A novel post-processing technique has been developed for reducing the effects of rotational motion artifacts. The rotation is constrained to be within the imaging plane and around the slice selection axis. The correction algorithm is based on a data corruption model in the spatial frequency space and a robust inverse filter using a conditioned singular value decomposition (SVD). The algorithm does not require a priori knowledge on the motion. Methods of estimating the motion parameters are incorporated. The performance of the algorithm has been verified using simulated motion. It has also been demonstrated to be robust against the presence of noise.

I. INTRODUCTION

POPULARITY of MRI over other imaging disciplines depends largely on its high spatial resolution and soft tissue contrast. In order to achieve such high quality, diagnostically interpretable images, relatively long image acquisition times are required, making it vulnerable to physiological or patient motion induced artifacts. In two-dimensional Fourier imaging [1], the artifacts present themselves predominantly as blurring or ghost repetitions of the moving structures in the phase encoded direction [2].

The techniques that have been proposed for reducing artifacts fall into two major categories. Some techniques attempt to prevent the motion from corrupting the data whereas the others use signal processing techniques to restore the image, incorporating the information embedded in the corrupted data (i.e. post-processing techniques).

The motion artifact prevention techniques depend basically on restricting the acquisition times (e.g. gating [3], pseudo-gating [4], ROPE [5]) or restricting the motion of the patients. The latter usually succeed in preventing extensive translational and rotational motions, yet require active co-operation from the patient. It was observed that small degrees of rotation can occur in spite of restraint measures, which are more effective at limiting nodding motions than rotations in the transverse plane [6].

Post-processing techniques are useful in restoring images corrupted by slight motion which are immune to motion prevention techniques. A number of such techniques have been proposed, which assume motion models such as translational motion [7]-[10], rotational motion [6][11] and expansion [12]. The translational motion model has aroused the greatest interest, due to the fact that gross translational motion is space invariant, and the artifact effects can be modelled using linear sets of equations [8][10]. In contrast, the rotational and expansion models have attracted relatively less attention due to their highly non-linear effects. Nonetheless, there are some generalised correction methods proposed, which either require pulse sequence changes [13] or exhaustive knowledge on the motion [14]. The latter uses an iterative correction via inverse problem solving and assumes that the corrupted image is close enough to the original image for convergence of the algorithm. This technique can be vulnerable to noise and errors embedded in motion measurements. We believe that individual motion models with motion estimation methods can help in overcoming the above deterrent effects. The corrected information on the motion and the image can then be used in conjunction with the above mentioned iterative technique to achieve stronger convergence properties and higher image quality.

A novel technique for reducing rotational motion artifacts is introduced in this paper. The rotation is assumed to occur between views and constrained to be within the imaging plane and around the slice selection axis. An investigation on the performance of the algorithm on gradual and sudden rotations has been conducted, to cover most of the possible scenarios. Gradual rotations are represented by rotations at constant angular velocity, where as the sudden rotations are represented by sudden single step rotations. Section 2 outlines the highly non-linear nature of the problem, together with a novel concept of spatial frequency space based rotation model involving the spatial frequency shift in the phase encoding direction. Section 3 describes a two step image reconstruction algorithm, for which, the singular value decomposition (SVD) [11] is used for stable computation. Section 4 introduces the technique of estimating angular velocity from noisy, corrupted signal, for gradual rotations. Section 5 and Section 6 present methods for estimating rotation view and
rotation angle respectively, for sudden rotations, followed by simulation results of the performance of overall strategy.

II. PROBLEM FORMULATION

In 2DFT MRI methods, the acquisition of a row of data is called a view. The data are spanned through the spatial frequency co-ordinates $U$ and $V$ in the frequency encoded and phase encoded directions respectively.

Given the transverse magnetisation $m(x, y)$ of the object, the detected signal is proportional to,

$$S(U, V) = \int \int m(x, y)e^{-i2\pi(Ux+Vy)}dx\,dy$$

(1)

discounting the relaxation effects. From Equation (1), it is quite apparent that $S(U, V)$ is equivalent to the 2D Fourier Transform of $m(x, y)$. For the purpose of deriving a discrete model of the detected MRI signal, the data are sampled with $N$ points in both $x$ and $y$ directions.

If the new co-ordinates, due to rotational motion of the object at an arbitrary time $t$, is given by

$$\tilde{x}(t) = x \cos \theta(t) + y \sin \theta(t)$$

(2)

$$\tilde{y}(t) = -x \sin \theta(t) + y \cos \theta(t)$$

(3)

the corrupted MRI signal $S'(U, V)$, due to time varying magnetisation can be mathematically formulated to be

$$S'(U, V) = \sum_{x=-N/2}^{N/2-1} \sum_{y=-N/2}^{N/2-1} m(\tilde{x}, \tilde{y})e^{-i2\pi(Ux+Vy)}$$

(4)

For almost all types of motion, it is reasonable to assume that the object is moving sufficiently slowly that it is effectively stationary during an acquisition of a single view [8]. Accordingly, the motion is assumed to occur only between views.

Due to patient restraint measures, the rotational motion can be limited to slight rotations. Thus, it is assumed that the total rotation of the imaged object is limited to approximately $35^\circ$.

According to the Fourier rotation theorem, a rotation of the imaged object by an angle $\theta$ produces an equal rotation of its Fourier transform or the spatial frequency spectrum [5]. The mathematical implication of this theorem on Equation (4) is given by,

$$S'(U, V) = S(\tilde{U}, \tilde{V})$$

(5)

where,

$$\tilde{U} = U \cos \theta + V \sin \theta$$

$$\tilde{V} = -U \sin \theta + V \cos \theta$$

For rotational motion with a constant angular velocity [11], the displacement of data in spatial frequency domain is graphically represented in Figure 1, and Figure 2 illustrates the displacement of data for sudden single step rotation. The skewed lines indicate the displaced positions of the data. It is assumed that the frequency encoded $U$ direction is sampled at an extremely fast rate, due to the continuous nature of these lines.

Figure 1: Distortion in the U-V space due to rotation at constant angular velocity.

Figure 2: Distortion in the U-V space due to sudden single step rotation.

It can be observed from Figure 1 and Figure 2, that the effect of rotational motion on the spatial frequency domain, is simply a shift in the phase encoded $V$ direction.
Figure 3 presents the locations of spatial frequency data for a single view, for both the rotated and unrotated cases.

![Figure 3: Illustration of shift along the V direction in the spatial frequency domain.](image)

The new location of data is given by $f(V)$, via mathematical derivation of spatial frequency shift $\Delta V$:

$$f(V) = V + \Delta V = \frac{(V - a \sin \theta)}{\cos \theta}$$  \hspace{1cm} (6)

A new motion model can be implemented to represent the artifacts due to rotational motion, using the spatial frequency shift. The modelled data $S'(U, V)$ are given by

$$S'(U, V) = \sum_{x=-\frac{N}{2}}^{\frac{N}{2}-1} \sum_{y=-\frac{N}{2}}^{\frac{N}{2}-1} m(x, y)e^{-i\frac{2\pi}{N}(Ux + f(V)y)}$$  \hspace{1cm} (7)

where $f(V)$ is given by Equation (6) and $\theta$ is the angle of rotation at view $V$.

III. IMAGE RECONSTRUCTION

The image reconstruction algorithm is based on the motion model given by Equation (7). Due to rapid sampling along the frequency encoded direction, it is assumed that $S'(U, V)$ values for any $U' = a$ are available via interpolation of acquired data, which are denoted by $\hat{S}'(a)(V)$. The reconstruction of the image is performed in two consecutive steps given by,

Step 1:

$$[S_d(y)] = T^{-1}[\hat{S}'(a)(V)]$$  \hspace{1cm} (8)

where $T$ is an $N$ by $N$ complex matrix, defined by,

$$T_{xy} = e^{-i \frac{2\pi}{N} xy(V)}$$

for $-\frac{N}{2} \leq y \leq \frac{N}{2}$ and $-\frac{N}{2} \leq V \leq \frac{N}{2}$

The matrix $T$ is often found to be ill-conditioned. The method of singular value decomposition (SVD) [15] was found to have the best performance in matrix inversion. By setting a lower limit to the smallest non-zero singular value, the SVD algorithm not only protects the solution from diverging, but also handles the problems of roundoff errors, finite precision of calculations and inherent noise in the corrupted signal $S'(U, V)$.

The symmetric properties of matrix $T$ can be used to reduce the computational burden. In gradual rotations, both variables $y$ and $f(V)$ were found to be symmetric around $y = 0$ and $V = 0$, which resulted in an 8-fold reduction in computations [11]. In sudden rotations, only $y$ was symmetric, effectively resulting in a 2-fold reduction in computations.

Step 2:

$$m(x, y) = \sum_{a=-\frac{N}{2}}^{\frac{N}{2}-1} S_d(y)e^{i \frac{2\pi}{N} ay} = F^{-1}\{S_d(y)\}$$  \hspace{1cm} (9)

$S_d(y)$ is defined as the shift corrected spatial frequency values for some arbitrary $U' = a$, where $-\frac{N}{2} \leq a \leq \frac{N}{2} - 1$. $F^{-1}\{\}$ denotes the inverse Fourier transform.

IV. ESTIMATION OF ANGULAR VELOCITY

The gradual rotations involve comparably low angular speeds (i.e. in the order of $10^{-3}$ r/s). The error which may be introduced in estimating such a small value can become significantly high [11]. Therefore, instead of estimating the angular velocity ($\omega_c$) directly, it is possible to estimate the value of a constant $C$, which is given by Equation (10).

$$\frac{\omega_c}{C} = \frac{\pi}{C}$$  \hspace{1cm} (10)

The value of $C$ is usually large (i.e. of the order of $10^3$), and thus, the associated estimation error may amount to be negligible [11].

Usually, prior to the estimation of $C$, through iterative or computational technique, it is essential to define a function which indicates the correctness of the reconstructed image, that in turn indicates the accuracy of the $C$ value, which has been used for the reconstruction process. The popular choice for this function would be to use the energy outside the region of support (ROS) of the image [10]. Unfortunately, the presence of noise complicates the use of this obvious choice, as this area will be usually dominated by noise. The pixel values outside the ROS are comparably smaller to the noise values and this region is most affected when confronted with any type of noise. The region inside the ROS, contains reasonably larger pixel values compared to the noise and is least affected.
The task of estimating \( \omega_c \) is further complicated due to the fact that various values of \( C \) usually produce different amplifications on the noise component in the final image, specially when \( C \) is not in the vicinity of the correct value. It was observed that when \( C \) value is largely deviated from its correct solution, the noise effects dominate the accuracy of the reconstruction, whereas at the vicinity of the accurate \( C \), the noise effects become fairly constant. This rather important result leads to a method of separating the two deteriorating effects on the image that stem from noise and the choice of \( C \). As explained previously, the region inside the ROS is comparably insensitive to noise and when the noise component is constant, the region outside the ROS can indicate the accuracy of the chosen \( C \) value. Thus, a function that incorporate both the energy outside and inside the ROS is the best choice for an error function, which indicates the accuracy of the reconstructed image. Such a function can be simply defined as,

\[
E = \frac{\text{Energy outside ROS}}{\lambda (\text{Energy inside ROS})} \tag{11}
\]

where \( \lambda \) is an appropriate optimising parameter, which has been arbitrarily set to 1.0.

Figure 4: Definition of \( y \)-strip inside and outside ROS.

Image reconstruction process involves \( 2N \) matrix inversions and a single 1D Inverse Fourier Transform. Each calculation of error function \( E \) is also expected to involve a similar number of calculations. To reduce the computational burden at the \( C \) value estimation stage, a novel idea of \( y \)-strip energy consideration is introduced [11].

4 illustrates how the \( y \)-strip inside and outside ROS are defined.

The energy inside and outside ROS can now be approximated, using only the \( U \) values around \( U = 0 \), where \( S'(U, V) \) is least affected by noise.

\[
E_{out \ ROS} = \lambda_1 \left[ \sum_{U = \frac{N}{2} - 10}^{\frac{N}{2} + 10} \left( \sum_{y = i}^{y = i+10} |s_U(y)|^2 \right)^2 \right] \tag{12}
\]

\[
E_{in \ ROS} = \lambda_2 \left[ \sum_{U = \frac{N}{2} - 10}^{\frac{N}{2} + 10} \left( \sum_{y = i}^{y = i+10} |s_U(y)|^2 \right)^2 \right] \tag{13}
\]

where \( \lambda_1 \) and \( \lambda_2 \) are constants.

The above computations involve only a few number of matrix inversions and no Inverse Fourier Transforms. The error function can now be defined by,

\[
E = \log E_{out \ ROS} - \lambda \log E_{in \ ROS} \tag{14}
\]

where \( \lambda = 1.0 \).

\( E \) goes through a global minimum at the desired value of \( C \), due to \( E_{out \ ROS} \) being minimised while \( E_{in \ ROS} \) being maximised.

V. ESTIMATION OF ROTATION VIEW

For the purpose of this paper, only inter-view motion is considered. Therefore, the rotation occurs in between the acquisition of data corresponding to two consecutive views. The view immediately prior to the rotation is denoted by \( V_{rot} \). The collected data, on the view \( V_{rot} \) and the preceding views are uncorrupted. As MRI data represent the spatial frequency domain of the image, each data component is expected to possess symmetric properties commonly found in the 2D Fourier Transform. However, since the data are acquired separately for each view, there may be some intensity variations among the signals corresponding to various views. The embedded signal noise also contribute to the disruption of symmetry. Nevertheless, it was observed that there is a detectable inconsistency among the magnitudes of the data, at the vicinity of the view at which the rotation occurs. A comparison between each \( (i) \)th and \( (N - i) \)th views yield a set of mismatch values which can be used to identify the threshold view \( (V_{th}) \) where the data mismatch first occurs. The data mismatch (MM) is defined by,

\[
MM(V) = \sum_{U = 0}^{N} \left| |S'(U, V)|^2 - |S'(N - U, N - V)|^2 \right| \tag{15}
\]

where, \( 0 \leq V < \frac{N}{2} \).

The function \( MM(V) \) has the maximum negative gradient between \( V = V_{th} \) and \( V = V_{th} + 1 \), which is being
used to identify the location of view \( V_{th} \). Therefore, \( V_{th} \) is given by,

\[
V_{th} = \{ V : \max( M \cdot M(V) - M \cdot M(V + 1)) \}
\] (16)

The \( V_{th} \) value indicate two possibilities for the rotation view \( (V_{rot}) \). The relationship between \( V_{th} \) and \( V_{rot} \) is given by,

\[
V_{rot} = V_{th}
\]

or

\[
V_{rot} = (N - V_{th} - 1)
\]

depending on whether the rotation occurs before or after \( V = \frac{N}{2} \) respectively.

In order to resolve the ambiguity and converge to a unique solution for \( V_{rot} \), one of the following steps was used, depending on the location of \( V_{th} \).

Step A:

If \( V_{th} < \frac{3N}{8} \), a fast image reconstruction was performed using Fast Fourier Transform (FFT), utilising a data window containing only the low frequency components of \( S'(U, V) \) for \( \frac{3N}{8} \leq U \leq \frac{5N}{8} \) and \( \frac{3N}{8} \leq V \leq \frac{5N}{8} \). If the resulting image was rotated, \( V_{rot} = V_{th} \); otherwise, \( V_{rot} = N - V_{th} - 1 \).

Step B:

If \( V_{th} \geq \frac{3N}{8} \), a fast image reconstruction was performed using Fast Fourier Transform (FFT), utilising a data window containing only the low frequency components of \( S'(U, V) \) for \( \frac{3N}{8} \leq U \leq \frac{5N}{8} \) and \( \frac{3N}{8} \leq V \leq \frac{N}{2} \). For \( \frac{N}{2} < V \leq \frac{5N}{8} \), the data are filled using,

\[
S'_{real}(U, V) = S'_{real}(N - U, N - V) \quad \text{and} \quad S'_{imag}(U, V) = S'_{imag}(N - U, N - V).
\]

If the resulting image is rotated, \( V_{rot} = V_{th} \); otherwise, \( V_{rot} = N - V_{th} - 1 \).

![Figure 5: Estimation methodology of the rotation view for sudden single step rotation](image)

In order to decide whether the resulting image was rotated, pixels with significant intensity, outside the region of support were detected. The intensity threshold can be varied according to the signal to noise ratio (SNR) of the data. If the number of pixels exceed a pre-defined threshold value, the image was judged to be rotated. The pre-defined threshold is usually small and has a maximum value of approximately 10.

The flow diagram in Figure 5 illustrates the steps involved in estimating the rotation view \( (V_{rot}) \) for sudden single step rotation.

VI. ESTIMATION OF ROTATION ANGLE

Once the \( V_{rot} \) is found via the methodology described in Section 5, the only other parameter necessary for image reconstruction is the rotated angle \( \theta_{rot} \). The technique of estimating this parameter involves the use of an error function based on the energy outside the region of support. The form of the error function is given by,

\[
E = \log [\text{Energy outside ROS}]
\] (17)

Only a relative figure of merit for the error is required, in order to determine the angle which is associated with the least error. Therefore, only low spatial frequency components around \( U = 0 \) can be used for the error calculations. An additional advantage is that the low frequency components are least affected by noise and are most reliable. The energy outside the region of support can be mathematically expressed using the \( y \) strips illustrated in Figure 4.

\[
E_{rot \ ROS} = \left[ \sum_{U=0}^{U_{lim} - 1} \left( \sum_{y=0}^{(y=0) \text{ to } y=0} |S'(y)|^2 + \sum_{y=1}^{N+1 - n_{buff}} |S'(y)|^2 \right) \right]
\] (18)

The above computations involve only a few number of matrix inversions and no Inverse Fourier Transforms. The summation limits on \( (t) \) and \( (t+1) \) are as defined by Figure 4, whereas \( U_{lim} \) and \( n_{buff} \) are decided according to the noise content. An increase in \( U_{lim} \) effectively increases the number of matrix inversions, yet resulting in a more accurate estimation of error. The value of \( n_{buff} \) is set to exclude corrupted high frequency components from each set of data. If the data contain high degree of noise, it is recommended to set a higher value for \( n_{buff} \) in order to reduce noise triggered inaccuracies in the estimate of the error.

VII. SIMULATION RESULTS

Rotation with constant angular velocity:

It is reasonable to assume that the maximum rotation angle can be restricted to some value using patient restraint measures. Accordingly, \( C \) value, described in
Section 4, will be contained within a known range. \( C \) can be traversed through this range in large steps to compute corresponding error function \( E \) value. The resulting \( E \) values can be used to reduce the correct \( C \) value range for the consequent iterations. It was observed that two consecutive \( C \) sweeps can converge to the desired \( C \), with reasonable accuracy. The sensitivity of the algorithm to the value of \( C \) can be controlled through the choice of condition number threshold in SVD process [15]. The value of \( C \) to the nearest integer is adequate for successful reconstruction of the final image [11]. Figure 6 illustrate the performance of the proposed algorithm in tracking the correct value of \( C \), and consequent image reconstruction.

\[
\begin{align*}
(a) & \quad (b) \\
\text{Error} & \quad \text{Error} \\
-3.50 & \quad -3.50 \\
1.00 & \quad 1.00 \\
0.50 & \quad 0.50 \\
0.50 & \quad 0.50 \\
C \times 10^3 & \quad C \\
0.50 & \quad 0.50 \\
0.50 & \quad 0.50 \\
0.50 & \quad 0.50 \\
0.50 & \quad 0.50 \\
\end{align*}
\]

Figure 6: Image reconstruction for MRI data corrupted by rotational motion at constant \( \omega_c = 0.00483 \text{ rad/s} \) (total rotation angle of 35.4°) and Gaussian white noise at 37 dB SNR: (a) estimation of \( C \) using a large range and large steps, (b) second iteration with smaller steps, (c) the corrupted image (MSE = 481.731), (d) reconstructed image (MSE = 33.651).

Sudden single step rotation:

The rotation view was found via detecting the maximum negative gradient in the mis-match function, and subsequently following Step A or Step B, as described in Section 4. The value of \( \theta \) was traversed through the pre-defined range in large steps, to compute the corresponding error \( E \) value. Further iterations were performed in the vicinity of the lowest error value, in order to converge to the global minimum. The value of \( \theta_{rot} \) was found, accurate to the nearest degree. The accuracy level of the estimation process was adequate for successful image reconstruction. The reconstructed image showed a high degree of motion artifact suppression, while preserving most image information.

Figure 7 illustrates the performance of the proposed algorithm in tracking the rotation view and subsequent estimation of rotation angle. The corrupted image and reconstructed image are also included, in order to demonstrate the degree of motion artifact suppression.

\[
\begin{align*}
(a) & \quad (b) \\
\text{MSE x 10^3} & \quad \text{Angle} \\
1.00 & \quad 1.00 \\
0.50 & \quad 0.50 \\
0.50 & \quad 0.50 \\
0.50 & \quad 0.50 \\
0.50 & \quad 0.50 \\
\end{align*}
\]

Figure 7: Image reconstruction for MRI data corrupted by sudden single step rotation (by a 15° angle at the 70th view), and Gaussian white noise at 35 dB SNR: (a) Estimating \( V_{rot} \) via magnitude mismatch, (b) Estimating \( \theta_{rot} \) via error minimisation, (c) the corrupted image (MSE = 950.045), (d) reconstructed image (MSE = 43.280).

VIII. CONCLUSION

A new algorithm to suppress rotational motion artifacts in MRI has been presented. New motion models for rotation at a constant angular velocity and sudden single step rotation have been developed, based on the frequency shift in the spatial frequency domain. The image reconstruction algorithm consists of two steps involving
the correction for spatial frequency shift and consequent inverse Fourier transform. The matrix inversion calculations were performed using conditioned singular value decomposition (SVD), in order to achieve stability against ill-conditioning and noise effects. The computations involved in the reconstruction process has been significantly reduced using the symmetrical properties of the motion models. Methodologies to estimate motion parameters were also incorporated, based on error functions involving both the energy inside and outside the region of support. The successful performance of the image reconstruction algorithm has been proven, in the presence of noise, using simulation results.

REFERENCES


Chaminda Weerasighe was born in Colombo, Sri Lanka. He received the B.E. degree with first class honours in Electrical Engineering from University of Wollongong, Australia. He is also the recipient of the University Medal in 1993. His thesis project on Adaptive Active Power Filters was judged the best thesis and was awarded the staff prize in 1993. He has also won the best student prize from IEE in 1991 and the IEE medallion in 1993. He was also awarded IEAust Sydney division medal in 1992 and IEAust Electrical College award in 1993. At the annual general meeting of the NSW Electricity Supply Engineers Association (ESFA), he was awarded the ESFA prize for 1992. In his undergraduate years, he presented a research paper On the merits and demerits in using Fast Fourier Transform in estimating harmonics in power systems, at the 1993 Australian Universities Power Engineering Conference (AUPEC '93) and was awarded the best student paper prize by the organisers, Pacific Power.

He has been attached to the Pacific Power Energy Efficiency Research Center and the State Engineering Corporation in Sri Lanka, before embarking on his Ph.D research, with the Imaging Group at the Department of Electrical Engineering, University of Sydney, where he has been awarded the AUSTA Electric prize for the best theoretical research project in 1996.
Application of Spiral Architecture to Edge Detection for Object Recognition

Xiangjian He and Tom Hintz
School of Computing Sciences
University of Technology, Sydney
PO Box 123, Broadway 2007
Australia
{sean, hintz}@socs.uts.edu.au

Abstract
Object matching has many potential applications in industry, defense and medical science. There have been many matching algorithms proposed in the past years. Most of these algorithms are based on edge detection. In this paper, our approach is to apply the Spiral Architecture with mathematical tools such as differential geometry and differential invariants. This allows us to parallelize the feature detection, and represent and analyse certain aspects of image information at the earliest processing stages of a machine vision system.

1 Introduction
A fundamental problem of object recognition is recognising objects within a given scene. Object matching, which measures the degree of similarity between two object sets that are superposed upon one another, plays a key role in the area of object recognition. In the past years, many matching algorithms were proposed, such as Chamfer matching [1], Borgefors matching [2] and the Hausdorff distance [3]. Based on the contours formed by the edge points, feature-based methods for the recognition of objects independent of their position, size, orientation and other variations have been the goal of much recent research (see, for example, [4-8]). Finding efficient invariant features is the key to solving this problem. To that end, contours extracted from images have proved quite useful in many object recognition algorithms. The extracted contour is then parameterized to provide a compact representational form for the geometric information, from which object identity may be determined. These algorithms begin with detecting the edge points.

The Spiral Architecture for Machine Vision described by Sheridan in [9] is a tool for the MIMD implementation of the earlier stage of edge points detection, known as 'segmentation'. Segmentation is the process of subdividing an image into its constituent parts. The purpose of the process is to reduce the complexity of the image. An image may be regarded as a collection of pixels with corresponding light intensities. An image of moderate size might contain one million pixels. Each pixel might have one of a number of possible intensities. Hence, the number that represents the number of possible patterns from such an image would be extremely large. Suffice to say there is no machine ever imagined that could cope with that kind of computational complexity. Consequently, there is an overwhelming need for a process that will reduce the complexity. This requires Segmentation. There have been many different types of segmentation algorithms. The segmentation as considered in [9] is a low level process that partitions the basis of light intensity.

Using the output of 'Segmentation', we would be able to apply the mathematics tools such as differential geometry to represent and analyse certain aspects of image information at the earliest processing stages of a machine vision system. An approach that has been advocated by Koenderink[10] and his co-workers describes image properties in terms of differential geometric descriptors, i.e., different combinations of derivatives. For example, the edge of an image is a set of points for which the gradient magnitude assumes a maximum in the gradient direction. So, the edge detection can be made on the basis of local directional derivatives. It is essential to base the analysis on descriptors that do not depend on the actual coordinatization of the spatial and intensity domains. Therefore, it is natural to require the representation to be invariant with respect to primitive transformations, like translations, rotations, scale changes, and certain intensity transformations. That is, the representation has certain types of differential invariants. This coincides with the invariant feature-based method for object recognition.

The context of this paper is organised as follows. Section 2 raises the question which leads to the consideration of edge detection of object recognition. The segmentation using Spiral Architecture is introduced in Section 3. In Section 4, detailed edge point detection by applying the tool
of differential geometry is described. We conclude in section 5.

2 Feature-based object recognition

Object matching plays a key role in the area of object recognition. It has many applications in industry, defense and medical science. Many matching algorithms were introduced in the past years. In this section we give a brief introduction of the work on the feature-based object recognition (see, for example, [7-8]) which leads to the consideration of edge detection.

The key of object recognition described by Tieng and Boles [7] is to construct an object representation, which is invariant under any affine transformation. This can be done by using the wavelet transformation of the parameterised object contour at different resolution levels with a pair of wavelet functions associated with a given scaling function [7,11]. At a particular resolution level, we use $I_j^h$ to denote the invariant representation of a given object $h$.

The main task of the recognition of an object is to recognize the object in an image as one of a number of models whose representations defined above are known. The representation of the object is matched with one of those in the database. An object in a scene is considered to belong to a specific class if the degree of dissimilarity between the object and models of that class is the smallest in comparison to the others. In order to measure the degree, we define a dissimilarity function below.

Let $f$ and $g$ denote the candidate model and unknown object, respectively. The dissimilarity function which measures the degree of difference between $f$ and $g$ at a particular resolution level $j$ can be defined as

$$d_j(f,g) = |I_j^f - I_j^g|.$$ 

This function requires that the two representations, $I_j^f$ and $I_j^g$, must have the same number of resolution levels. This requirement can be obtained as shown in [7].

It is possible that the different objects can have the same representation at a particular resolution level but not at all resolution levels. Thus, in order to increase the matching accuracy, $d_j(f,g)$ must be performed at several resolution levels, say, $K_j (j = 1, 2, \cdots, L)$ where $K_1$ and $K_L$ are the preselected finest and coarsest resolution levels used in matching, respectively. The overall dissimilarity, $D$, can be simply defined as the average of $d_j(f,g)$ at these $L$ levels, i.e.,

$$D(f,g) = \frac{1}{L} \sum_{j=1}^{L} d_{K_j}(f,g).$$

The problem now is how to choose the resolution levels for the matching process. It is well known [7] that the magnitude of wavelet coefficients of a signal at each resolution level is proportional to the distributed energy of the signal at the associated frequency band. Since our representation is constructed based on these coefficients, its magnitudes at each level are also related to the distributed energy. As a result, the resolution levels where most of the signal energy is distributed can be determined. By using the representation at these levels in matching, the effect of noise on the representation is significantly reduced.

In practice, based on the representations of models in the database, one can choose the resolution levels where most of their energy is concentrated. Having determined the range of these resolution levels for each model, we select levels which are common to the ranges and we name them the 'Common Levels'. The representations at the Common Levels are used in the classification process. This gives the unknown object an equal chance of matching with the candidate models in the database.

3 Segmentation

Segmentation is the process of subdividing an image into its constituent parts in order to reduce the complexity of the image. The resulting subdivisions must be meaningful and the process should embody an organizing principle that reflects the innate organization of the visual world.

There have been many different types of segmentation algorithms devised to operate under a variety of conditions. In this section, we introduce a segmentation algorithm using Spiral Architecture.

3.1 Spiral architecture

An image may be considered as the collection of pixels (picture elements). These elements correspond to the position of the photo receiving cells of the image capturing device. In the case of the human eye, these elements would represent the relative position of the rods and cones on the retina. The geometric arrangement of cones on the primate's retina can be described in terms of a hexagonal grid. This leads to the consideration of an image as the collection of hexagonal cells as displayed in Figure 1.

Each of the individual hexagons is labelled with a unique address as described in [9]. This is achieved by describing a process that begins with a collection of seven hexagons. Each of these seven hexagons is labelled consecutively with addresses 0, 1, 2, 3, 4, 5 and 6 as displayed in Figure 2.
The repetition of the above steps permits the collection of hexagons to grow in powers of seven with uniquely assigned addresses. It is this pattern of growth that generates the Spiral. Furthermore, the addresses are consecutive in base seven.

The important aspect of the hexagonal grid is that it possesses special computational features that are pertinent to the vision process. The hexagonal grid can be described in terms of the mathematical object known as a Euclidean ring [9].

### 3.2 Segmentation algorithm

The usual segmentation algorithms are in either the category of partitioning on the basis of grey-level discontinuities or the category of partitioning on the basis of grey-level similarities. The segmentation algorithm presented here falls into both categories. The algorithm is named 2DSA (short for Two Dimensional Segmentation Algorithm) [9]. It has a number of distinctive features. For example, the algorithm embodies a high degree of parallelism, proceeds concurrently and is intrinsically linked to the Spiral Architecture.

#### 3.2.1 Notation

Let $S$ represent the entire image space (the collection of all hexagons, each of which is identified by a unique address). The segmentation algorithm partitions $S$, in accordance with a Connectedness relationship $C$, into subspaces $S_1, S_2, \ldots, S_n$, each of which is a collection of some hexagons. The relationship $C$ forms equivalence classes $S_1, S_2, \ldots, S_n$, i.e., every hexagon in the same class is 'connected' to each other and the hexagons belong to different classes are 'disconnected'. A disconnection between adjacent addresses will be made on the basis of the significance of the differences in grey-levels at the address concerned. The significance of the change in grey-level across two
adjacent pixels, \( g_1 \) and \( g_2 \) which are assumed to be in the range 0 to 255, is defined by [9] 

\[
\delta(g_1, g_2) = 256 - \max\{g_1, g_2\} + (513 - |g_1 - g_2|) \cdot |g_1 - g_2|/2.
\]

One may find that distinct pairs of grey-levels give rise to a unique \( \delta \) and that for a given absolute difference in grey-level, the \( \delta \) is inversely proportional to the magnitude of the individual grey-levels. This approach imitates a characteristic of human vision. That is, grey-level changes at low high intensities are more noticeable than equal changes at higher intensity values.

The notation defined here will be employed to describe 2DSA as follows.

### 3.2.2 Local segmentation

We first describe the procedures that operate at the ‘cluster’ level, which is a collection of seven hexagons as shown at Figure 2.

**Compute \( \delta \) values** The first procedure to consider is to compute the \( \delta \) values for each of the twelve adjacency relations in the cluster using the known grey-level values.

**Fission** The second procedure is a process of disconnection. One first disconnects the pair of addresses which returns the biggest \( \delta \) value. Then further disconnections are made to resolve any contradictions to the laws governing equivalence relations.

**Fusion** The third procedure is a fusion process. This process assigns single Class Value to all hexagons which are in the same equivalence class. The Class Value should be the address of one of hexagons in the equivalence class.

### 3.2.3 Global segmentation

The three local procedures are extended so that an image composed of more than one cluster can be segmented. The global activity is governed by a synchronization process that coordinates the local activity.

Suppose that there are \( m \) clusters for an image. Then, the synchronizing process communicates with and controls the processing at each of the \( 7^m \) (one for each hexagon) processes, \( H_i \) (\( i = 1, 2, \cdots, 7^m \)). The processing associated with each hexagon is the combined processing of the three local procedures. The local processes or procedures are essentially sequential. The global processing, on the other hand, is a collection of processes that operates concurrently at all resolutions. Figure 5 is a diagram of the aforementioned processes.

![Figure 5: A flow diagram of the segmentation program.](image)

**Remark** [9] Three points regarding 2DSA can be stated. Firstly, it is independent of the illumination conditions. Secondly, it indicates how the special adjacency properties of clusters can be exploited computationally. Thirdly, it shows how Spiral Architecture can be employed in a parallel environment as a useful data structure for one of the low level image processing operations essential to machine vision.

### 4 Edge detection

Computer vision is a cross-disciplinary field with research methodologies from several scientific disciplines. The Scale-space Theory is proposed by Lindeber [13] for how certain aspects of image information can be represented and analysed at the earliest processing stages of a machine vision system. This theory can be used for edge detection on the basis of segmentation described above.

#### 4.1 Scale-space representation

Let \( f : \mathbb{R}^2 \rightarrow \mathbb{R} \) be a signal of an image. The scale-space representation \( L : \mathbb{R}^2 \times [0, \infty) \rightarrow \mathbb{R} \) is defined such that the representation at ‘coarser scales’ is equal to the original signal, i.e.,

\[
L(\cdot; 0) = f(\cdot), \quad (4.1)
\]

and the representation at ‘coarser scales’ is the convolution

\[
L(\cdot; t) = g(\cdot; t) * f(\cdot),
\]

where \( g : \mathbb{R}^2 \times (0, \infty) \rightarrow \mathbb{R} \) is the Gaussian kernel

\[
g(x, y; t) = \frac{1}{2\pi t} e^{-\frac{(x^2 + y^2)}{2t}}.
\]

In fact, \( L \) is the solution of the isotropic diffusion equation

\[
\partial_t L = \frac{1}{2} \Delta L = \frac{1}{2}(\partial_{xx} + \partial_{yy})L, \quad (4.2)
\]

with the input image \( f \) taken as initial condition, i.e., (4.1).

Scale-space representation is used to suppress and remove unnecessary and disturbing details so that later stage processing tasks can be simplified.
This can be explained in that the signal which is $L$ becomes gradually smoother as $t$ increases. The equation (4.2) gives a direct physical interpretation of the smoothing transformation. The scale-space representation can be understood as the result of letting an initial heat distribution $f$ evolve over time $t$ in a homogeneous medium. Hence it can be expected that fine-scale details will disappear, and images become more diffuse when the scale parameter $t$ increases.

4.2 Edge detection in continuous case

A natural way to define edges from a continuous grey-level image $L : \mathbb{R}^2 \times \mathbb{R}^+ \rightarrow \mathbb{R}$ is as the set of points for which the gradient magnitude assumes a maximum in the gradient direction [13]. To give a differential definition of this concept, introduce a curvilinear coordinate system $(u, v)$, such that at every point the $u$-direction is parallel to the gradient direction of $L$, and at every point the $v$-direction is perpendicular to the $u$-direction. Moreover, at any point $P = (x, y) \in \mathbb{R}^2$, let $\partial_u$ denote the directional derivative operator in the gradient direction of $L$ at $P$ and $\partial_v$ the directional derivative operator in the perpendicular direction. Then at $P$ the gradient magnitude is equal to $\partial_v L$, denoted by $L_{\theta}$, at that point. Assuming that the second and third order directional derivatives of $L$ in the $v$-direction are not simultaneously zero, the condition for $P_0$ to be a gradient maximum in the gradient direction may be stated as

$$L_{\theta} = 0 \quad \text{and} \quad L_{\theta\theta} < 0. \quad (4.3)$$

Note that,

$$\partial_u = (\cos \beta) \partial_x - (\sin \beta) \partial_y,$$
$$\partial_v = (\cos \beta) \partial_x + (\sin \beta) \partial_y,$$

where $(\cos \beta, \sin \beta)$ is the normalized gradient direction of $L$ at $P_0$. Hence, (4.3) is equivalent to

$$\tilde{L}_{\theta\theta} = L_{\theta\theta}^2 - 2L_{\theta x} L_{\theta y} + L_{\theta y} L_{\theta y} < 0,$$
$$\tilde{L}_{\theta\theta\theta} = L_{\theta\theta\theta} + 3L_{\theta x} L_{\theta x} L_{\theta x} + 3L_{\theta y} L_{\theta y} L_{\theta y} < 0.$$

By reinterpreting $L$ as the scale-space representation of a signal $f$, it follows that the edges in $f$ at any scale $t$ can be defined as the points on the zero-crossing curves of $\tilde{L}_{\theta\theta}$ for which $\tilde{L}_{\theta\theta\theta}$ is strictly negative. Note that with the above formulation there is no need for any explicit estimate of the gradient direction.

4.3 Discrete approximation

Given discrete data, let us investigate the effect of approximating derivatives [13] of $L$. Here, discrete approximations to $\tilde{L}_{\theta\theta}$ and $\tilde{L}_{\theta\theta\theta}$ will be computed by using numerical approximation [13]. Since zero-crossing curves in the discrete case do not exactly exist, it is implemented by considering each cell with four neighbouring points in the image,

$$\{(x, y), (x + \epsilon, y), (x, y + \epsilon), (x + \epsilon, y + \epsilon)\}, \quad (4.4)$$

where $\epsilon > 0$ is sufficiently small, and performing a two-step linear interpolation as follows. For any pair of adjacent points of those in (4.4) having opposite sign of $\tilde{L}_{\theta\theta}$, introduce a zero-crossing point on the line between, with the location set by linear interpolation. Then, connect any pair of such zero-crossing points within the same cell by a line segment. If this operation is performed on all cells in an image, then the edge segments are obtained and can be easily linked into polygons by an edge tracker to form the edge.

5 Conclusion

In this paper, we apply the Spiral Architecture and Differential Geometry to the edge detection of an image, which is required on the feature-based method for the object recognition. Firstly, we segment the given image in the low level using Two Dimensional Segmentation Algorithm. This can be implemented concurrently. Then, we detect the edge of the image, at any scale, by the scale-space representation.

Acknowledgement

The authors would like to thank Phil Sheridan for useful discussion.

References


Visualization of Data Dependency Analysis in Parallel Program Design

E. Trichina and B. Thomas
Advanced Computing Research Center
University of South Australia
The Levels, SA 5095
Australia

J. Oinonen
Department of Computer Science
University of Joensuu
P.O.Box 111, 80101 Joensuu
Finland

Abstract

Our project explores how to use three dimensional graphics, interactive animation, interface metaphors and different information visualization techniques to aid human comprehension of the complex phenomena a programmer has to deal with in the process of designing parallel algorithm and programs. The goal of the project is to construct and integrate with existing systems graphical editors to help users visualize complex concepts comprising the parallel program development process. In this paper we report on the visual analysis tool, which uses a three dimensional model to describe the data dependencies to the computational domain and to visualize transformations of this domain relevant to the main steps in parallel program design.

Keywords: parallel program development, algorithm analysis, data dependency graph, visualization, user interface metaphors, animation.

1 Introduction

One of the pioneers in visual formalisms Prof. David Harel wrote in his paper [9] published in 1988:

We are entirely convinced that the future is "visual". We believe that in the next few years many more of our daily techniques and scientific chores will be carried out visually, and graphical facilities will be far better and cheaper than today. The languages and approaches we shall be using in doing so will not be merely iconic in nature (e.g., using the picture of a trash can to denote garbage collection), but inherently diagrammatic in a conceptual way, perhaps, also three-dimensional and animated. They will be designed to encourage visual models of thinking and tackling systems of ever-increasing complexity, and will exploit and extend the use of our own wonderful visual system in many of our intellectual activities.

Parallel program design and analysis is one of such complex activities, where many difficulties stem from the principle inadequacy of pure textual formalism to specify parallelism in an understandable fashion. This is caused by the inherent sequentiality of any textual formalism.

Visualization is being seen as an excellent way to aid human comprehension of complex phenomena with large quantities of data, parallel processes and their communication patterns.

In this paper we outline our ideas of how to improve the parallel software development process by integrating 3D graphics, animation, direct manipulation techniques, and user interface metaphors into parallel programming environments. These ideas lay a foundation for a two-years project, which is carried out in the School of Computer and Information Science, The University of South Australia.

The idea behind our project is the unification of the graphical user interface and the application. There are two aspects to be considered. Firstly, the user interface and the application must be more tightly coupled, i.e., the user interface needs more "semantic feedback" from the application. Second, there should be a single environment for the development of both.

We attempt to fulfill this idea through an interactive visual system which aims at allowing the user to represent mathematical equational models directly in the system. The system performs dependency analysis of equations to extract parallelism and automatically generate parallel code for numerical solutions.

2 Visualization in Parallel Programming

Parallel programming is notoriously difficult. Each parallel computer requires a long period of learning and gaining experience. The overhead for developing the first parallel program prevents potential users from taking advantage of the advances in hardware development. Nowadays, low-priced parallel machines or ready-to-use networks built from usual workstations provide a low-cost access to parallelism. This class of machines and their
potential which motivates our work: to provide a step towards the broader acceptance of parallel computers by non-expert in parallel programming.

2.1 Parallel Program Design

The complexity of parallelism is inherent: designing a parallel program comes up with more sub problems than in the sequential case. The main steps of a parallel program development are briefly described below.

Step 0. The design starts with the Algorithm Specification. Usually mathematical formulae, logics or some special notation is used at this level. However, sometimes it might be a “normal” sequential program (e.g., nested loops). No references to parallelism is usually expected at this level.

Step I. The specification is analyzed to explore a potential parallelism. To find which computations can be carried out independently from each other, i.e., in parallel, the input/output relationships between computations are explicitly identified. The result of the Algorithm Analysis usually is a Data Dependency Graph (DDG) which is known to be a fruitful starting point for parallelization.

Step II involves an analysis of how suitable the DDG is for a specific parallel hardware, and choosing an appropriate parallelization strategy. Different parallel architectures usually require different models of parallelism and parallelization strategies. To tailor the DDG to suit parallel computer architectures, syntactical and/or semantical Transformations can be applied to the DDG to transform it to a new graph, which can be easily implemented in hardware. The outcome of this level is a logical structure of the program, a so-called Process Graph, where all parallel components and communications between them were identified and exhibited.

Step III. During the Code Generation stage, source code for each process in a process graph is derived. This includes also code which describes the spatial structure of the process graph, communication between processes, and synchronization.

Step IV. It is necessary then to find a good assignment of logical processes and communication channels to physical processors and links depending on the nature of the process graph and of the computer architecture. This step is known as Code Mapping.

Step V. Performance Analysis is the speed-up and efficiency analysis and performance tuning of the final parallel program.

2.2 Visualization in Parallel Programming Environments

The use of two dimensional graphics in parallel programming is not new [4]. Graphical methods were found especially useful when describing:

- Static configurations like the hardware topology of multiprocessor systems, the concrete network of a cluster of workstations (interconnection, I/O ports, mass storage access) or the logical structure of a static process system. Figures describing hardware structure are very common in papers, books and manuals [11]. However, very few programming environments represent such figures explicitly.

- Inter-process structures, i.e. coarse grain process structures using uniform communication patterns (e.g., data-parallel systems). A number of graphical extensions to parallel programming languages have been implemented [17] [2], [8], [1], where the various facets of a parallel program are given as annotations to a process graph. These tools are based on the hypothesis that a hybrid specification methodology, which integrates the graphical specification of parallel aspects and the textual specification of sequential aspects, is the best possible basis for an integrated programming environment for parallel programming.

- Intra-process structures, i.e. detailed control structure of a process and its interactions with other processes and fine-grain parallelism [23], [16].

- Mapping parallel program onto underlying parallel hardware architecture. This error-prone and specially difficult step in parallel program development requires user’s knowledge of computer architecture in details. Some mapping tools provide this knowledge in a visual form [12], [15].

- To present results of performance analysis and for animating parallel software execution, where the communication information is extracted and a number of visual displays are produced and animated with the streams of events to drive the simulation [1], [7], [18].

As one can see, all the efforts have been concentrated on Steps III – V of parallel program development. In other words, visual tools were mainly intended for parallel code design and debugging rather than for exploration of parallelism in a task specification.

However, the first two steps, an analysis of the algorithm and “tailoring” data dependencies to suit computer architecture, are of paramount importance. Many times non uniform or poorly defined communication or synchronization have made creation of correctly functioning and efficient software a very expensive process. A number of mathematical methods and “know-how” techniques exist to help on these initial stages of parallel program
development [5], [24], [11], [13], [22]. However, mathematical methods are often too involved to be used by an average programmer; to appreciate know-how techniques and pick up the "right one", the user has to already be an expert in parallel programming.

This is where we believe three dimensional interactive graphics will be an unqualified success. The third dimension provides an extra degree of freedom for conveying information. Using 3D graphics, we can visualize all the data dependencies evolving in time as a three dimensional graph, displaying it in a screen. Visual abstraction will shift information to the perceptual level to help assimilate mathematical notions.

To illustrate our point and to facilitate the following description of the graphical system, we consider a set of typical problems the programmer faces during algorithm analysis and parallelization. Then we show how visualization facilitates the comprehension of the parallelization techniques.

3 An Example: an Algorithm Analysis and Parallelization

Suppose, we need to design a parallel program suitable for a pipeline of 
processing elements (PE), where each PE has its own local memory, CPU, and a limited number of fast communication links connecting it with its neighbors in a pipeline. This is a typical computer organization for distributed memory MIMD computers, and thus, a realistic assumption. As an example, we consider a typical problem in signal processing: a finite impulse response filter.

3.1 Initial specification

The finite impulse response (FIR) digital filters are used in a variety of ways including speech transmission, coding theory, and image processing. The FIR filters are linear shift-inversion systems, whose region of support is finite; for many cases, the FIR impulse response \( a_k \) is non-zero only in the limited area, \( 1 \leq k \leq n \). The FIR handles sample points of \( x(t) \), namely \( x_1, \ldots, x_n \). Thus, sets of \( n \) sample points and impulse responses are given and a resulting set of \( n \) points is produced. The FIR can be specified by the following system of recurrent equations:

\[
\begin{align*}
g[i, 0] &= 0, 1 \leq i \leq n; \\
g[i, k] &= g[i, k - 1] + a[k] \cdot x[i + k - 1], \\
&1 \leq k \leq n, i + k - 1 \leq n; \\
g[i] &= g[i, k], 0 \leq i \leq n, i + k - 1 = n.
\end{align*}
\]

This set of equations can be considered as a formal specification of the problem at hands, representing Step 0 in a parallel program design.

3.2 Data dependency analysis

Next step is to analyze the specification in order to understand:

- what potential parallelism this method contains;
- how suitable this parallelism is for the class of parallel computers we are dealing with (or, in other words, how potential parallelism can be exploited taking into account limitations and constrains of the target hardware);
- to choose a parallelization strategy allowing an efficient implementation.

To perform an analysis, one has to trace all data dependencies specified by the method [24]. A naive explanation of this step is that data dependencies impose a partial order on computations, i.e., when two computations are connected by the input-output relationship (i.e., the result of one computation is the input value for another one), these two computations must be performed in the order imposed by this relationship. On the other hand, computations which do not depend on each other can be carried out simultaneously.

An analysis of the mathematical specification is not easy, especially for a person who is not yet used "to think parallel". Of course, observing the equations, one notices, that in order to compute the value of \( y[i, k] \) the value of \( y[i, k - 1] \) has to be computed first.

Thus, \( y[i, k] \) and \( y[i, k - 1] \) cannot be computed in parallel. To understand what can, requires even more efforts. After spending some time "unfolding" the system of equations, and writing something like

Figure 1: Data Dependency Graph for FIR filter
\[ y[1, 1] = y[1, 0] + a[1] \times x[1 + 1 - 1] \\
y[3, 1] = y[3, 0] + a[1] \times x[3 + 1 - 1] \\
\ldots \\
y[1, 2] = y[1, 1] + a[2] \times x[1 + 2 - 1] \\
\ldots \\
\ldots \\
\]

(and may be even depicting the results in a form of a graph), one comes to understand the data dependency structure, and realizes, that there is a great deal of potential parallelism in this method. For example, for every fixed recurrent step \( k \), computations of all \( y[i, k] \), \( 1 \leq i \leq n \), are independent from each other and in principal can be computed in parallel, provided that element \( a[k] \) can be accessed by all these computations simultaneously. Thus, each recurrent step can be completed in one time unit, and, if all potential parallelism could be exploited by the hardware, the computations for \( n \) sample points would take only \( O(n) \) time units.

3.3 Tailoring potential parallelism to hardware requirements

However, the limitations and constraints of the hardware should be taken into account while designing a practical and efficient parallel algorithm. For example, since shared read is not possible in the target architecture (each processing element has direct access only to its local memory), the straightforward implementation of the parallel algorithm is hardly possible. The additional questions we have to answer now are:

- how to distribute all computations along the pipeline of processing elements;
- how to distribute data so that all processing elements will have them when needed;
- how to ensure that there will be no bottlenecks and/or deadlocks;

and many others. As one can see, the discovery of the potential parallelism is not enough. The problem is to find a suitable parallelism, and this requires a great deal of work. The lessons we have learned from our previous experience [21], and which we tried to enhance with this brief example are:

1. Generation of Data Dependency Graphs at Steps 0 - 1 of parallel program development is important for understanding and exploring potential parallelism. This is a difficult cognitive process which requires meticulous tracing of input/output dependencies between computations.

Pure textual presentation hides these relationships and can hardly provide any insight on this process.

2. Potential parallelism can not always be exploited in a straightforward manner; transformations "tailoring" data dependencies to suit computer architecture (such as elimination of an unlimited access to a memory location) are often needed. A good deal of experience is required to understand what would be a desired transformation, and how it affects the input/output relationships between computations.

3. Transformations of a Data Dependency Graph into a Process Graph at Step II must be well understood. It is important to choose appropriate transformations and to estimate their effect on the design and efficiency of the result.

We believe that the conceptual difficulties in comprehension of parallelization techniques stem from principal inadequacy of purely textual formalisms to specify the problem so, that the parallelism can be expressed in an understandable fashion. Indeed, one often resorts to drawings in order to understand the data dependency structure of the algorithm, and at later steps to describe a spatial structure of the Process Graph. In the rest of the paper we intend to show that using visualization is not only natural, but often necessary to perform an initial analysis of the specification and to come up with a suitable parallelization method.

4 3D Graphics and Animation and Parallelization of the Algorithm

One of the aims of the research is to investigate, design, implement, and test a three dimensional graphical editor to support initial steps of parallel program analysis and development. This editor provides a comprehensive system interface which makes the problems of algorithm analysis and tailoring a potential parallelism to suit computer architecture, a more intuitive and effective process.

4.1 3D Graph Editor

Our goal is to construct a graphical system which uses a three dimensional model to describe the problem. This system will be a centerpiece for problem specification, analysis and transformations. The objective of the proposed 3D editor is to support a concise abstraction of input/output relations, to increase the homogeneity of Data Dependency Graphs, and to provide immediate feedback from algorithm’s transformation which will lead to more effective acquisition of mathematical theories and parallelization strategies. This editor provides tools that do the following:
• analyze computation index space defined by a problem specification and generate the corresponding computational domain\(^1\);

• generate Data Dependency Graph in this index space as follows:
  - associate computations with the lattice points in the 3D domain;
  - associate edges with input/output relations between computations;

• visualize generated Data Dependency Graph in a 3D space;

• scale, rotate, zoom the graph to exhibit all data dependencies in order to facilitate algorithm analysis;

• perform an animation of transformations on graph structures, such as elimination of non-local dependencies via pipelining, and projection of the computational domain onto the domain of processing elements. Visualize results of these transformations as new graphs.

Such visual abstractions can shift information to the perceptual system and facilitate understanding of formal aspects of corresponding transformations. In particular, details can be viewed in context, making them more meaningful;

• help perception of a work-load using colors and color intensity. For example, if graph nodes are non evenly colored, it means that its nodes will perform non even tasks;

• visualize the wavefronts of parallel computations using color propagation.

These tools allow programmers to work in the spatial and visual domain (graphical editors) and oppose to a symbolic and cognitive domain (textual programming languages). The spatial and visual domain allow programmers use natural strengths of humans, such as pattern matching, information abstraction, and problem decomposition. These tools will not replace in total the need to work in the symbolic and cognitive domain, but provide a means to work on a problem in different information spaces.

To illustrate this thesis, we describe the scenario of parallel program development for the same task, where design steps are visualized and animated.

4.2 Visual aid in data dependency analysis and parallelization

As we already mentioned, at Step 0 we use pure mathematical formalism (i.e., a system of affine recurrence equations) for initial problem specification. A syntax-oriented interactive editor was specially designed for the purpose of communication with the user, who is supposed to specify her problem in a precise mathematical form. Our syntax-oriented, pull-down menu-driven editor prompts the user to input required information in a form that is concise with the “usual” mathematical notation.

4.2.1 Visualization of the Data Dependency Graph

At Step 1 the specification has to be analyzed in order to explore potential parallelism. Using 3D graphics makes this analysis a more intuitive process. The domain of computations defined by this set of equations is a two dimensional integer coordinate space; two sets of matrix indices comprise two dimensions; the second dimension is allocated for unfolding the levels of recurrence.

To explore an arithmetic parallelism, the graphical environment transforms the set of recurrent equations into an explicit Data Dependency Graph by tracing the associated index space and using proper arcs to display the input/output dependencies in the computation domain. This graph is visualized in a separate window, as shown in Fig.1.

The advantage of the visualization for Step I of parallel program development is obvious: just one look at Fig.1 tells the programmer that the problem is characterized by a regular data dependency structure, and contains potential parallelism. However, further visual analysis reveals that data dependencies are somewhat not suitable for a pipeline of processor elements. To exaggerate this, the editor allows to “bounce” a node; all the adjacent

---

\(^1\)Initially, we restrict our research only to specifications in a form of affine recurrence equations and/or nested DO loops.
edges follow the node, creating a visual effect of global broadcasting as depicted in Fig. 2.

These manipulations with the graph indicate that there are features contradicting the constraints of the target parallel hardware, namely,

- some dependencies are not local, moreover, they are not even a constant distance away,
- there exists a shared read of some variables on each level of recurrence (one node sends an information to a number of nodes, which is a parameter of the size of the problem);
- the number of outgoing edges for some nodes is not constant, and is a parameter of the problem size;
- Data Dependency Graph is two dimensional, while the target architecture is one dimensional. This means that an additional analysis and transformations are required to find an optimal mapping of the 2D index domain of the problem onto 1D domain of processing elements.

4.2.2 Visualization of transformations

At Step I the Data Dependency Graph has to be transformed in order to suit computer architecture. Rotating, scaling and "slicing" the graph may help to come up with the idea how to eliminate such undesirable features as shared read and non-local dependencies by explicit pipelining.

For affine recurrent equations, our system provides an option, where appropriate transformations can be selected (using a special menu) from the set of transformations provided by the environment. Transformations for affine recurrent equations, including elimination of shared read, and various transformations for graph regularization are well-known, and have clear mathematical formulation. They can be carried out by the system automatically.

The results of the transformation are visualized. Fig. 3 illustrates the general layout of the screen when the transformation for elimination of shared read has just been applied. From the system menu two items were selected, namely Tools, containing a set of high-level procedures relevant to parallel program development techniques, and Effects, which is a menu of 3D manipulations on a given graph. From the Tools menu, the option transform was chosen, and a transformation for elimination of shared read has been applied to the original DDG automatically yielding a new graph, depicted in the 3D View window. To check the effect of the transformation, one of the nodes with all adjacent edges was highlighted, and the text specifying new computations in this node was displayed on a screen.

4.2.3 Visualization of domain morphisms

The problem a programmer is facing now is how to project the two dimensional computation domain onto a pipeline of processing elements.

The problem of domain morphisms is well studied in the literature, and many mathematical methods exist to describe such transformations [5], [13], [21]. However, sometimes mathematical methods are too involved, too demanding for an average programmer to use them freely. Often, an idea of what the final planar design might be is needed even more than rigorous methods, because learning "to think parallel" takes some time, and at this stage it is important to understand what actually stands behind mathematical formulae and affine transformations.

We found that standard rotating of the Data Dependency Graphs in a 3D space is an unqualified success in this respect. Indeed, viewing the DDG
from different view points imitates projection of a 3D or 2D figure onto correspondingly 2D or 1D space.

As previously, for affine recurrent equations the environment provides a transformation, which projects three dimensional data dependency graphs onto a planar domain of processing elements.

The advantage of visualization is obvious: by standard 3D graphical editors, manipulations the users can explore different possible designs, compare them and select the one they think is the best.

Compare, for example, the results of two different possible projections, depicted in Fig. 4 and Fig. 5. Obviously, the Process Graph in Fig. 4 is "better": it has fewer nodes, and more regular structure, with all links going into one direction, while the graph depicted in Fig. 5 has almost a double number of nodes and bidirectional links.

All possible projection directions are listed in the Projection menu, and choosing one of the projections, the user decides upon the structure of the parallel program. As we said, the actual graph transformation is supported by the system, and can be carried out automatically. This transformation is animated, i.e., the graphical display illustrates how nodes and edges of a 3D graph are "collapsing" into one node. At the end of the animation, the resulting Process Graph is displayed on a screen. From now on, the user may use all sort of tools provided by traditional parallel programming environments to write actual code.

The 3D editor provides a possibility for the user to work on a problem in different information spaces. Observing different projections, programmers may use their imagination and a natural pattern matching skills and abstraction ability to "re-invent" the desirable transformations. Even if the environment will not support an actual (textual) generation of the corresponding Process Graphs, projection metaphors depicted in Fig. 4 and 5 provide the user with an insight what the suitable process graphs for the problem might be and how to achieve these.

A user can scale, rotate and zoom graphs to get a better view or better layout. Also, a programmer may zoom a node in order to edit or/and view an annotation, which can be a (formal or informal) specification of computations carried out by this node, or a piece of code.

4.3 Dynamic Information

Lon Barfield, Eddy Boeve and Steven Pemberton made the observation in [3], that humans are more sensitive to moving objects than static ones, and that this development should be exploited in user interfaces. An improvement to current interfaces is to change the transitions between state information in an interface. Standard state information has a "before and after" nature, the View System [3] provides a general system-wide way to add "fill-in" animations between this state information. These fill-in animations supply continuity for the end user, to better convey the change in state of the interface. We used the same technique in our system as well, most noticeable in the visualization of the transformations of the DDG into a Process Graph. Worth noting, that we also designed two new graphical interface metaphors, which we describe below.

4.3.1 Bounce

The bounce metaphor animates one or more nodes to determine connectivity between those nodes and other nodes in a graph. The animation is to have the node oscillates from its position to a position of a specified plane, as shown in Fig. 2. It appears to work best with an individual animated node, and this gives clear visual feedback as to the nature of the connections (i.e., data dependencies between computations).

The animation allows the user to obtain visual information in a smooth and pleasing fashion. If the node were to jump from the two extreme positions of the animation with no in-between steps, the effect would be a jarring snap between two grossly different images. The user is momentarily startled; a small but significant disruption occurs before the new appearance of the screen is digested and work can proceed. With experience, users learn to ignore the disruption, but some of its effect remains. The overall impression is of an abrupt, hard-edged interaction that is jarring and tiring.
4.3.2 Fan in-out

As in the bounce metaphor, the fan in-out metaphor can animate the process of data dependency analysis by helping to determine connectivity between computations represented as nodes in a graph. Instead of moving the nodes, the center of the arcs are pulled away from their current position forming a “V” shape, as shown in Fig. 6. The animation is also oscillates to give the user continuous visual feedback of the connectivity of the nodes. We believe this metaphor is best suited to the situation of viewing many nodes at once. The fan in-out metaphor is less disruptive to the overall screen.

![DDG: fan in-out metaphor](image)

4.4 Hierarchical Information

The work of George G. Robertson, Stuart K. Card, and Jock D. Mackinlay[14], which has pioneered the area of 3D information visualization, indicates that the 3D nature of the world can be a powerful visual cue for keeping track of hidden information. Their thesis is that by arranging the information in 3D space and smoothly changing the view, people can better keep track of where information is located, even if it is not currently visible.

When we first started viewing the DDG’s in 3D space, we noticed that the amount of information on the screen was overwhelming. The overload was already intolerably high with a graph of only 55 nodes. It became apparent that the information had to be reduced to produce representations, which could be quickly comprehended. Paramount was that the topological structure of the presented information had to be retained. The solution we choose was to *deemphasize non-relevant information*, to reduce the total amount of information presented, and to *highlight* the information for closer inspection, as described below.

4.4.1 Transparent objects

Information is deemphasized by making the nodes more transparent or smaller. This is similar to the selective dynamic manipulation of visualizations techniques, described in [6], where they dimmed color and shrunk size of graphical objects to make information of less relevance less visible. To highlight information, the height, width and color of the nodes were adjusted to change resolution of the information space.

Our work is concerned with 3D nodes and 3D links. We have borrowed some of the above ideas of color and size changes to help with visualization of a subset of a total information space. We have added animation and transparency to the set of visual cues. There are three metaphors for interrogating 3D information spaces: bounce, fan in-out, and slicer. In addition, we resort to the usage of color, size, and text to further strengthen the visualization process.

4.4.2 Slicer

To gain a better understanding of connectivity between DDG’s nodes, a slicer metaphor has been used. A user can denote an axes (or a set of axes) in a Cartesian space, along which he/she prefers to view a DDG, and only nodes and edges corresponding to the selected axes will be visible. The user can then “walk” along one of the axis highlighting nodes and corresponding links one by one. This strengthens the notion of nearest neighbors, and allows to “build up” a structure of the entire DDG incrementally.

Another important advantage of the slicer is that the user can investigate dependencies implied by different variables separately, and thus, to locate shared read and non-local variables.

4.4.3 Color

To reduce the total amount of information presented to the user, non-relevant information is made slightly transparent. This allows for important information to be easily accessed, while the topological relation of the important information to entire information space is retained. The amount of transparency can be adjusted by the user for individual taste.

Information concerning the length of communication links and amount of computations carried out by an individual node can be highlighted by brighter colors. These colors serve only as means of drawing the users attention to a particular problem; other means of more precise depiction of this information are also available³. Bright and high

³For example, such as viewing the source code of the computations associated with any selected node.
contrasting colors can grab the attention of a user quickly.

4.4.4 Size

Size is used in the same fashion as color to hide information. The non-relevant information can be reduced in size to the point of disappearing. To totally remove some forms of information is sometimes a preferred option, but we have to be careful to preserve the overall structure of the information domain. For example, if the nodes of the DDG were to be removed, the edges should remain. The structure of DDG then remains through the end points of the links.

In a Process Graph, which is obtained as the result of the projection of a DDG onto processor domain, some nodes have to carry out more computations that others. In this case, size is used to inform the user about the load balance: if all nodes have the same size, the amount of computations carried out in these nodes is equal. In the case of one large node, that node would have a large amount of computation relative to the rest of the graph.

4.4.5 Text

Finally, the computations to be performed in each of the nodes can be presented in a textual form. The text is placed on the screen next to the node containing it, as shown in Fig. 7. This enables the user to mentally connect the text with a particular node. Textual specification of computations in multiple nodes can be displayed at once for comparison of computational load.

However, displaying text in a straightforward manner obscures the picture. Often textual fragments overlap, and a visual "structure" of information is lost. Probably, we should employ a complementary strategy, such as "zoom" a selected node, and then to view the corresponding text, which in this case may be written inside the node. Another option may be a combination of size and transparency while displaying text: for example, initially, text may be printed in a very small font, and displayed in semi-transparent boxes alongside the corresponding nodes. In this case, we will need to use a "magnifying lens" metaphor to view the text.

5 Technical Aspects of Implementation

We decided to use SGI's Open Inventor software toolkit for the project. Open Inventor is a 3D graphics object library written in C++. It implements a number of basic and more sophisticated features required when dealing with three-dimensional graphics and animation. When needed, the user can derive new classes from the Inventor classes, thus implementing features that don't exist in the core of the toolkit. Open Inventor file format has been the base for VRML, the language used in WWW for describing three-dimensional objects.

The nicest thing about Open Inventor is that it gives basic 3D visualization "for free": the program only has to define the locations and relationships of different objects in the 3D space and the Inventor system does the rest. After the shape of an object is defined, no extra code is required to rotate the object in the 3-D space and look at it from different directions.

Open Inventor as such is independent of any underlying window system. However, the workstation on which the software development was carried out had Motif windowing system installed, so it was used for some of the user interface elements, mainly menus, because Open Inventor doesn't have direct support for these.

The existing system has a fully graphical, menu-driven user interface. Its main features can also be activated from the command line. Via the user interface, user has extensive control over program behavior. A comprehensive HTML-based context sensitive on-line hypertext help system has also been fitted.

This far, the development of the software has taken about 4 man-months, and the total size of source files is some 8000 text lines. The program is written mainly in C++, some parts in C. For most of the task, Inventor classes were used as such. A small number of new classes were defined in order to be able to implement some of the visualization techniques used in the system.
6 Future Work

Our ultimate goal is to integrate our system into existing parallel programming environments and to enhance all the levels in parallel program development within the Integrated Graphical Programming Environment. The main problem we have to resolve is a continuity of tools.

While 3D graphs representing data dependencies do not necessarily have to be described in a user-friendly textual form (in other words, the internal representation of these graphs is an internal matter of the graph editor), because at that stage only visualization and direct manipulation with visual images is important to the user, the 2D projections of pipelined dependency graphs have to be described explicitly in a textual form. The reason is that 2D projections represent Process Graphs, i.e., graphs that describe a spatial structure of the parallel program.

Although it is possible to hide a textual representation of these graphs from users, leaving a direct graph manipulation as the only option for further steps of parallel program development (and thus, totally separating a spatial structure of the parallel program from “computation aspects”), we believe that providing tools for a concise and structural textual description of a spatial structure of the parallel program is much a better strategy.

Our intention is to integrate the graph description language GG [15] into a visual programming environment providing possibilities for both, visual and textual parallel program development at Steps III - VI. Moreover, the ultimate goal is to ensure an easy and coherent “switch” from a visual mode to a textual mode and vice versa.

7 Concluding Remarks

The significance of this experiment can be summarized as follows:

1. The use of three dimensional information spaces to help users to construct programs is an emerging new research area. The recent advances in technology for hardware and software allow fully rendered three dimensional information models to be applied to a wider range of applications. It is now practical to apply these recent technologies to the challenging problem of providing visual support and aiding human comprehension to inherently difficult problem of algorithm analysis and implementation.

As our experiment has proven, an introduction of three dimensional graphics, interactive animation and different information visualization techniques and interface metaphors aid a human comprehension of complex phenomena with large quantities of data and their dependencies, involved in the Algorithm Analysis.

Moreover, visualization of spatial and temporal relationships among computational activities is crucial for improving an acquisition of the ideas of possible parallelization strategies.

Indeed, it takes about two hours to view all possible projections for FIR filter. Since each projection corresponds to a Process Graph (and, hence, to some parallel program), the whole process of “parallelization” of FIR filter problem takes about half a day.

On the other hand, even for an experienced parallel programmer it would take at least half a day to perform one projection “manually”. This is where visualization is a big winner.

We also believe that visual interpretation of mathematical formulae reduces the disparity between the programmer’s conceptualization of the problem and its specification.

2. The project also contributes to visualization research involving technical and theoretical work. More specifically, the use of three dimensional information brings with it a new and interesting set of problems. Additional issues, which 3D graphics should help to investigate, are summarized below:

- What kind of interaction is appropriate for the three dimensional information space?
- How the three dimensional editor interacts with a conventional two dimensional interface?
- How much information can be displayed on a screen in any time so that human perception is not overwhelmed?
- How much information (and what sort of information) can be expressed by colors and/or their intensity?

3. Our research project is a part of efforts to use visualization in improving the software development process, improving software reliability, and facilitating the scalability and re-use of software.

The project addresses important issues in the areas of Parallel Programming Environments and Graphical User Interface. An immediate application of the outcome of the project can be seen in teaching parallel programming subjects in universities or industry.

References

In Parallel Computing: From Theory to Sound Practice, W. Joosen and E. Milgrom eds. IOS Press, Amsterdam, 92.


Visualising the Dynamics of Facial and Vocal Expression of Emotion

by

E. C. Willock

School of Computer Science and Engineering
University of New South Wales, Sydney, Australia 2052
<charlesw@cse.unsw.edu.au>

Abstract:

Understanding the emotional content of facial and vocal expressions is fundamental to understanding much of the real meaning in human communication. It is important in understanding the needs of children and partners, those in need of assistance, in assessing credibility, for biometric identification, through to correct interpretation of needs in international negotiations.

By transforming temporal image sequences and audio information to spatial form, human facial expressions and vocal expression can be visualised together. This provides a powerful observation and measurement tool for analysis and understanding of the co-expression of human verbal and non-verbal communication.

In a more general form computerised analysis of image sequences provides opportunities in analysis of medical imaging, understanding weather patterns and other remotely sensed data, understanding the interaction of pilots or industrial operators with their instrumentation, and offers opportunities for emotion-aware computers and more realistic and “considerate” robots.

Introduction:

This paper is a brief introduction to some of the many factors related to emotional expression in human communication, and one form of computer perception of emotional meaning through visualisation of the dynamics of facial and vocal expression. Here I focus primarily with visualisation of facial emotional expression since analysis and spatial representation of speech signals through such techniques as Fourier transforms, pitch diagrams etc. are well known [1]. “Stack-plotting”, introduced here, allows analysis of visual and speech co-expression of emotion.

Applications:

Understanding emotional content of facial and vocal expressions is important to determining the real meaning in human communication with applications including:

- making friends / pleasing the boss
- understanding children / partners
- assessing medical or psychiatric state
- detecting deception
- biometric identification
- business and international negotiations
- creating emotionally aware robots

Measuring Emotion:

Emotion is readily seen in

- Facial Expression (visual)
- Vocal Expression (verbal)
- Time series of physiological data

Visual Expression:

For visual examination of facial expression we can interpret the emotional meaning from different viewpoints:

- static
- dynamic
- masking
An example of interpreting the emotion from the static viewpoint would be attributing happiness to a photograph of a person smiling. At a dynamic level, an image sequence of that same person smiling could provide a better understanding of the interpretation of the emotion through analysis of the rate at which the smile became apparent, the time for which it was held and the return to neutral, or the decay from the peak of the emotion - the emotion now ascribed might differ from the original static interpretation. In certain circumstances, a safe or socially acceptable emotion (such as a smile) can be used to mask a socially unacceptable one (such as anger). By understanding facial expression at the static, dynamic and masking level, a much better interpretation and objective assessment of the emotional intention of the communicator can be achieved.

Vocal Expression:

Information about emotional state can also be inferred from vocal expression including:

- rate of delivery
- hesitation
- pitch

Physiological Parameters:

Emotion can also be inferred from physiological parameters. These include measurement of:

- pulse rate, and variation
- skin resistance and variation
- breathing dynamics
- hormonal changes

Pan-Cultural Expressions:

There is a huge range of terms for emotions. However, at present, only six are internationally agreed as being readily differentiated in all cultures. They are:

- Disgust
- Fear
- Anger
- Surprise

Added to that there is recognition for a face which shows no emotional expression...called the “Neutral” face.

Facial Expressions:

Any individual emotional expression is made up of movement of a number of (pairs of) facial muscles. Comprehensive description of the actions of individual muscles of the face is best obtained from the reference work “The Facial Action Coding System” [2].

The easiest way to begin to learn about facial expression is to practice moving particular facial muscles while looking in a mirror. The location of the muscles of facial expression can be found any general atlas of human anatomy (e.g. Snell [3], p400), while which muscles are involved in communicating particular emotions can be found in various Altases of facial expression [4] [5].

A valuable way of appreciating the activity of facial muscles is to place your fingers against the skin over the affected muscle. Those muscles which are activated can be felt as they activate in the expression.

Neural Pathways:

Two different pathways are thought to be involved in facial muscle activation. Voluntary muscle control, utilises the brain’s cortical motor strip and the pyramidal system [6]. Emotional expression arising sub cortically reaches the face via the extra-pyramidal system.

Expressions reaching the face via different pathways have different properties and thus, potentially, can be differentiated. Voluntary (pyramidal) system expressions are characterised by being:

- unsynchronised
• not smoothly co-ordinated
• asymmetrical
• not consistent (quick, slow or held)
• “controlled” in appearance

In contrast, emotional (extra-pyramidal) system driven facial actions are characterised by muscle movements which are:

• synchronised,
• smooth,
• symmetrical,
• consistent, and
• reflex or “ballistic” in appearance

**Personality Influences:**

Emotional response to a situation is, in large part, related to the personality of the individual and to the context of the communication. What angers, frightens, amuses, saddens, disgusts, can differ substantially between individuals. Likewise different personality types are known to be likely to tell different kinds of lies.

Potentially, some personality properties can also be extracted from facial expressions. For example, extraverts are known to be more facially active than introverts.

**Slice Stacking Technique:**

By transforming temporal image sequences and audio information to spatial form, human facial expressions and vocal expression can be visualised together.

extracting a temporal sequence stack of an eye closure.

A particular feature of the tool is the ability to display image information in a form which can be rapidly reviewed by a human researcher, and which can also be used as input for an inferencing engine.

In common with many visualisation techniques, stackplotting of image sequences [7] provides four important perceptions often at least partially concealed by traditional “statistical” signal processing or static measurement techniques:

• a global view of all of the data
• perception of values near each point
• indication of the value at each data point
• indication of effects of noise in the data

Through these attributes stackplotting can provide the following benefits:

• insight into spatial / temporal structure
• perception of patterns within the data
• perception of anomalies and noise
• clear identification of one-off events

**Figure 2.** demonstrates the stack obtained by placing a temporal sequence of slices together whereby showing the detailed dynamics of a blink (shown here for the left eye) taken from video information recorded at 25 frames per second. Immediately noticeable is the rapid upper lid closure followed by a slower opening. In contrast a wink would show slower onset and offset due to greater involvement of the muscles surrounding the eye (orbicularis oculi).
When in a more relaxed state the dynamics show a slower onset of closure which is still more rapid than the opening (Figure 3).

Multiple rapid repetition of blinks illustrated above may be observed when a person is surprised or disbelieving.

Key to the utility of the technique is that these image slices can be precisely synchronised with vocal expression and displayed adjacent to other image stacks, along with those of visualised speech, displays of signals, and graphical depiction of analysis etc.

Other Applications:

The techniques developed for analysis of image sequences of faces have application to other image sequences. They are for example applicable to:

- analysis of medical image sequences
- remotely sensed data [8] [9]
- operation of industrial instrumentation

Acknowledgements:

For their guidance, my thanks to my supervisor Associate Professor Paul Compton, Head of the School of Computer Science and Engineering at the University of New South Wales, and to Assistant Professor Mark Frank, School of Communications, Information and Library Studies, Rutgers University, my co-supervisor.

Computerised stackplotting and applications were originally developed in conjunction with Dr Richard Thompson and Patrick McIntosh. Their collaboration is gratefully acknowledged.

References


Two dimensional wavelet transforms

Patrick M. Lenders and Anne M. Sjöström

Department of Mathematics, Statistics and Computing Science
University of New England
Armidale - NSW 2351
pat@cs.une.edu.au

Abstract

The wavelet transform is a signal processing technique for applications such as non-stationary signal analysis and data compression. High compression ratios can be achieved while maintaining quality reconstructions.

We present a generalization of the Mallat pyramid for nD wavelets. We introduce a transformation to uniformize the equations defining the successive levels of the pyramid. The synthesis of wavelet functions allows us to explore the possible 2D Haar wavelet transforms and to study their properties. We propose a methodology for implementing these wavelet transforms in parallel architectures like systolic arrays. More specifically, we show that there is a perfect match between the wavelet algorithms and the MultiPhase multiRate Array (MPRA) architectures.

Keywords: VLSI synthesis, systolic arrays, multirate arrays, wavelets.

1 Introduction

The wavelet transform is a signal processing technique for applications such as non-stationary signal analysis and data compression. High compression ratios can be achieved while maintaining quality reconstructions.

In [8] Grochenig et al. present Haar bases for nD wavelets. In the present paper, we propose a methodology for implementing these wavelets in parallel architectures like systolic arrays.

The paper is organized as follows. The next section presents briefly the wavelet transform and Mallat’s pyramid algorithm (cf. [7]). Section 3 introduces the multiresolution analysis for the generalized Haar wavelet in the n dimensional case. In section 4 a fast algorithm for the computation of the nD wavelet transform is presented. This algorithm is a generalization of the Mallat pyramid. An implementation in systolic arrays is also presented. Finally section 5 outlines an example, and its implementation in different architectures.

2 The wavelet transform

The recently introduced wavelet transform (cf. [2]) is a tool that deals with non-stationary signals. The signal is decomposed by means of a special analysis function \( \psi \), called the basic wavelet, which is translated in time or space (for selecting the part of the signal to be analyzed) and then dilated or contracted by using a scale parameter \( \sigma \) (in order to focus on a given range of oscillations).

The one-dimensional wavelet transform is a function of two variables \( (t, \sigma) \). The wavelet transform is local in frequency and in space. The wavelet transform of a function \( f(x) \) is the set of values \( b_{i,j} \) such that:

\[
 f(x) = \sum_{i,j} b_{i,j} \psi(2^i x - j)
\]

The index \( i \) specifies the level of detail (larger \( i \) means finer details), and the index \( j \) specifies the location along the \( x \) axis. The index \( i \) corresponds to the frequency in the Fourier transform.

How can we obtain the coefficients \( b_{i,j} \)? Let \( a_{i,j} \) be the value of \( f(x) \) at level \( i \) and at loca-
tion $x = j/2^i$. The successive levels of detail can be obtained from the finest level of detail:

$$a_{i-1,j} = 2^{-1} \sum_k a_{i,j} C_{j-2k}$$

which is a low pass (LP) decimation filter. In this formula, the $C_i$ are constant coefficients. The values $b_{i,j}$ can be obtained from the values of $a_{i,j}$:

$$b_{i-1,j} = 2^{-1} \sum_k a_{i,j+k} (-1)^k C_{i-k}$$

which is a high pass (HP) decimation filter. Notice that the decimation factor is 2, and thus the input rate is halved after every level.

In $n$ dimensions, the decomposition is carried out by means of a set of basic wavelets $\psi_i$. The wavelet transform of a function $f(x)$ is the set of values $b_{i,j,k}$ such that:

$$f(x) = \sum_{i,j,k} b_{i,j,k} \psi_i(A^j x - k)$$

Notice that $x$ and $k$ are $n$ dimensional vectors and that $A$ is an $n \times n$ matrix. It can be shown that there exist $q - 1$ basic wavelets $\psi_i$ ($q = |\text{det}(A)|$).

3 \hspace{1cm} \textbf{Multiresolution analysis}

Wavelet bases can be constructed through a method called Multiresolution analysis (MRA). For a detailed presentation of the concept of MRA, please refer for example to [3]. To define a multiresolution analysis, we will use the notion of dilation. A linear transformation $A$ is a dilation if $A$ leaves $Z^n$ invariant and has eigenvalues $\lambda_i$ such that $|\lambda_i| > 1$. These properties imply that $|\text{det}(A)| = q \leq 2$. We will also use the notion of translation. The translator $\tau_y$ is defined by: $\tau_y f(x) = f(x-y)$.

A multiresolution analysis associated with $(Z^n,A)$ is an increasing family $\cdots \subset V_{j-1} \subset V_j \subset V_{j+1} \cdots$ ($j \in Z$) of closed subspaces of $L^2(R^n)$ with the following properties:

- $\bigcup_{j \in Z} V_j$ is dense in $L^2(R^n)$, and $\bigcap_{j \in Z} V_j = \{0\}$.
- $f(x) \in V_j$ if and only if $f(Ax) \in V_{j+1}$.
- $V_0$ is invariant under $\tau_\gamma$.
- There exists a function $\phi \in V_0$ (the scaling function) such that $\{ \tau_{\gamma \phi, \gamma} \in \Gamma \}$ is a complete orthonormal basis for $V_0$.

The wavelets $\psi_i$ and their translates form an orthonormal basis of $W_0$ (the orthogonal complement of $V_0$ in $V_1$). The wavelets can be built from the scaling function.

We will study a generalization of the Haar wavelet in 2 dimensions. In this case, the scaling function is the characteristic function of a measurable set $Q$. An acceptable set $Q$ must satisfy three constraints which are related to the dilation $A$. The following theorem defines these constraints.

**Theorem 3.1** Given a bounded measurable set $Q$ such that:

1. $Q \cap (Q + k) \simeq 0, \forall k \neq 0$ and $k \in Z^n$,
2. there is a collection of $q$ lattice points $k_i$ that are representatives of distinct cosets in $Z^n/AZ^n$ such that: $AQ \simeq \bigcup_{i=1}^q (k_i + Q)$, and
3. $\bigcup_{k \in Z^n} (Q + k) \simeq R^n$,

its characteristic function is the scaling function of a multiresolution analysis associated with $(Z^n,A)$. Conversely, if the scaling function of a multiresolution analysis associated with $(Z^n,A)$ is the characteristic function of a bounded measurable set $Q$, this set $Q$ satisfies the constraints 1, 2 and 3.

4 \hspace{1cm} \textbf{Implementation}

Let us assume that $\phi$ is a scaling function for an MRA associated with $(Z^n,A)$. We are interested in the case where the scaling function is the characteristic function of a measurable set $Q$. We will show how we can calculate the wavelet transform of a function $f(x)$ with a cascade of low- and high-pass filters. This algorithm is a generalization of Mallat’s pyramid algorithm.

If $f(x)$ is in subspace $V_r$, we can develop $f$ as follows:

$$f(x) = \sum_{k \in Z^n} a_{r,k}\phi(A^r x - k)$$
Multiplying by $\phi(A^r x - k)$ and integrating, we have:

$$a_{r,k} = q^r \int f(x) \phi(A^r x - k) dx$$

$$= q^r \int_{A^r x - k \in Q} f(x) dx$$

$$= f_r(k)$$

where we define $f_r(k)$ as the average value of $f(x)$ when $x \in A^{-r}(Q + k)$. We are interested in the 2D case ($n = 2$). At the finest level of detail (for example $r = 0$), a computer image is actually composed of pixels. Following [1], we define a greyscale image of finite resolution as a map $\mu : P \rightarrow [0, 1]$ where $P$ is the set of pixels and $\mu(k)$ is interpreted as the grey tone of the pixel $k$. We postulate that the function $\mu(k)$ is the function $f_0(k)$, the average of the function $f(x)$ (the image) for $x \in Q + k$. In other words, the shape of the pixel approximates the set $Q$. By using the definition of the MRA, we can calculate $a_{r-1,k}$ as a function of $a_{r,k}$:

$$f(x) = \sum_l a_{r,l} \phi(A^r x - l)$$

$$= \sum_k a_{r-1,k} \phi(A^{r-1} x - k)$$

$$+ \sum_{k',i} b_{r-1,k',i} \psi_i(A^{r-1} x - k')$$

Thus:

$$a_{r-1,k} = q^{r-1} \int f(x) \phi(A^{r-1} x - k) dx$$

$$= q^{r-1} \sum_l a_{r,l} \int \phi(A^r x - l)$$

$$+ \sum_{k'=1}^q \phi(A^r x - Ak - k_i) dx$$

$$= q^{-1} \sum_{i=1}^q a_{r,Ak+k_i}$$

In this formula, $r$ is the level of detail (larger values of $r$ correspond to more detail), $k$ is the vector defining the position, $q = |\text{det}A|$ is the number of cosets in $Z^n/AZ^n$, and $k_i$ are the lattice points that are representatives of the $q$ distinct cosets of $Z^n/AZ^n$. Thus the values of $a$ at level $r - 1$ are the averages of the values at level $r$. This corresponds to a low pass filter (cf. section 2).

Similarly, we can calculate the values $b_{r-1,k,i}$ ($i = 1, \ldots, q - 1$) as a function of $a_{r,k}$.

$$f(x) = \sum_l a_{r,l} \phi(A^r x - l)$$

$$= \sum_k a_{r-1,k} \phi(A^{r-1} x - k) + \sum_{k',i} b_{r-1,k',i} \psi_i(A^{r-1} x - k')$$

Thus:

$$b_{r-1,k,i}$$

$$= q^{r-1} \int f(x) \psi_i(A^{r-1} x - k) dx$$

$$= q^{r-1} \sum_l a_{r,l} \int \phi(A^r x - l) \psi_i(A^{r-1} x - k) dx$$

$$= q^{-1/2} \sum_{l=1}^q a_{r,Ak+k_i} \psi_i$$

This is a high pass filter (cf. section 2).

4.1 Example

The twin dragon bases can be defined from the dilation:

$$A = \begin{pmatrix} 1 & -1 \\ 1 & 1 \end{pmatrix}$$

and the two vectors:

$$k_1 = \begin{pmatrix} 0 \\ 0 \end{pmatrix}, \quad k_2 = \begin{pmatrix} 1 \\ 0 \end{pmatrix}$$

To verify that the characteristic function of the set $Q$ is a scaling function, we must verify that the three constraints of theorem 3.1 are satisfied (cf. [3]). We can calculate the value of the successive levels with the following formulas:

$$a_{r-1,k} = 1/2(a_{r,Ak} + a_{r,Ak+k_2})$$

and

$$b_{r-1,k} = 1/2(a_{r,Ak} - a_{r,Ak+k_2})$$

Figure 1 represents the output of the first level of the wavelet transform of the famous Lenna image.
4.2 Implementation in systolic arrays

The low and high pass filter formulas can be written:

\[ a_{-1}(Ap) = 1/q \sum_{i=1}^{q} a_{i}(Ap + k_i) \]
\[ b_{-1,j}(p) = q^{-1/2} \sum_{i=1}^{q} a_{i,j} a_{-i}(Ap + k_i) \]

These formulas are difficult to implement in systolic arrays because they are not uniform. Moreover, the low pass and high pass values are computed at the same location \( p \); thus it is impossible to store the new 'image' at the location of the old 'image'. Finally, these equations make pipelining of the successive levels very difficult. To remedy these problems, we can apply the transformation technique developed in [6]. We apply a domain transformation of the variable \( a_{-1} \):

\[ a_{-1}(Ap) = 1/q \sum_{i=1}^{q} a_0(Ap + k_i) \]
\[ a_{-2}(p) = 1/q \sum_{i=1}^{q} a_{-1}(A^2p + Ak_i) \]

....

It is easy to see that the first equation is uniform. To render the second equation uniform, we apply the domain transformation \( A^2 \) to the variable \( a_{-2} \):

\[ a_{-1}(Ap) = 1/q \sum_{i=1}^{q} a_0(Ap + k_i) \]
\[ a_{-2}(A^2p) = 1/q \sum_{i=1}^{q} a_{-1}(A^2p + Ak_i) \]

....

The general forms are (with \( j = 1, \cdots, q-1 \)):

\[ a_{-r}(A^r p) = 1/q \sum_{i=1}^{q} a_{-r+1}(A^r p + A^{r-1} k_i) \]  \hspace{1cm} (1)
\[ b_{-r,j}(A^r p) = q^{-1/2} \sum_{i=1}^{q} u_{i,j} a_{-r+1}(A^r p + A^{r-1} k_i) \]  \hspace{1cm} (2)

A few remarks pertain to these equations.

- We can deduce a simple location for the new values at every level: we can place the average \( a_r \) at location \( A^r p \) and the \( q-1 \) outputs of the high pass filters at locations \( A^r p + k_i \).
- We do not need the shape of the set \( Q \) (or \( A^r Q \)) to compute the transform, but of course we need it to interpret the meaning of the transform (i.e., the average of the signal over a set \( A^r Q \)).
- From theorem 3.1 (constraint 2), we deduce:

\[ A^r Q = \bigcup_{i}(A^{r-1}Q + A^{r-1} k_i) \]

Assuming that \( A^0 Q \) is the pixel at location 0, we can compute recursively the set \( A^r Q \).

In [5], we define a MultiPhase multiRate Array (MPRA) as a set of sparse, uniform, recurrence equations. Equations 1 and 2 are uniform (the dependency vector \( A^{r-1} k_i \) does not
depend on the position $p$) and sparse (there are 'holes' in the index space because the matrix $A^r$ is not the identity matrix), and thus we can expect a good match between this wavelet algorithm and the MPRA architecture.

The index space represents the space–time domain, which describes the activities of the different processing elements (pe) at successive clock cycles. From equations 1 and 2, we deduce that the points $A^r p$ of the index space are points where some computation is done. Moreover, the vectors $A^{r-1}k_i$ represent data dependencies: a datum must move from point $A^r p + A^{r-1}k_i$ to point $A^r p$. The following section illustrates these concepts with an example.

5 Example

The matrix:

$$A = \begin{pmatrix} 2 & 1 \\ 0 & -2 \end{pmatrix}$$

has a determinant equal to minus four. We can choose the four representatives of distinct cosets in $\mathbb{Z}^n / AZ^n$ as:

$$k_1 = \begin{pmatrix} 0 \\ 0 \end{pmatrix}$$
$$k_2 = \begin{pmatrix} -1 \\ 0 \end{pmatrix}$$
$$k_3 = \begin{pmatrix} -1 \\ 1 \end{pmatrix}$$
$$k_4 = \begin{pmatrix} -1 \\ -1 \end{pmatrix}$$

The corresponding fractal, along with a $2 \times 2$ square centered at the origin, is shown in figure 2.

The following implementations are directly derived from the general form presented in the previous section (cf. equation 2). As $q = |\text{det}(A)| = 4$, we have one low pass filter equation for $a_r(A^r p)$, and three high pass filter equations for $b_{-r,j}(A^r p)$. As these four equations are very similar, we will focus on the low pass filter equation in the following.

5.1 MPRA implementation

The data dependencies have the general form $A^r k_i$ (cf. equation 2). We can easily verify that all of the dependencies have a negative $i$ component. Thus if we choose this component as the time variable in our time-space representation of the algorithm, the causality constraint will be satisfied (cf. for example [6]). Indeed we have:

$$A = \begin{pmatrix} 2 & -1 \\ 0 & -2 \end{pmatrix}$$

$$A^2 = 4 \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$$

$$A^3 = 4 \begin{pmatrix} 2 & -1 \\ 0 & -2 \end{pmatrix} = 4A$$

Moreover, the products $Ak_i$ have a negative $i$ component:

$$Ak_2 = \begin{pmatrix} 2 & 1 \\ 0 & -2 \end{pmatrix} \begin{pmatrix} -1 \\ 0 \end{pmatrix} = \begin{pmatrix} -2 \\ 0 \end{pmatrix}$$

$$Ak_3 = \begin{pmatrix} 2 & 1 \\ 0 & -2 \end{pmatrix} \begin{pmatrix} -1 \\ 1 \end{pmatrix} = \begin{pmatrix} -1 \\ -2 \end{pmatrix}$$

$$Ak_4 = \begin{pmatrix} 2 & 1 \\ 0 & -2 \end{pmatrix} \begin{pmatrix} -1 \\ -1 \end{pmatrix} = \begin{pmatrix} -3 \\ 2 \end{pmatrix}$$

Thus all the products $A^r k_i$ have a negative $i$ component.

Figures 3, 4 and 5 represent the space-time domain of the MPRA implementation of the first three levels of the wavelet filter. The axis $pe$ is the linear array of processing elements, and the axis $time$ is the time dimension.

For clarity of the drawings, we include only the dependencies at the first three levels. The dependencies at the following levels have the
Figure 3: *Space-time* domain of the example (first level)

Figure 4: *Space-time* domain of the example (second level)
same slopes, but are longer. It should be clear that the same array can compute all the successive levels, as well as the output of the three high pass filters. The nodes with a circle (○) and a bullet (●) are active at level 1, and the nodes with a bullet (●) are active at level 3.

The wavelet transform is based on a dilation; thus the present (2D) example has a decimation along the two dimensions. Decimation along the time dimension means that some variables will travel through the array at slower speed. Decimation along the space dimension means that some processing elements will be more active than others. A real implementation will have to redistribute the load amongst neighboring processing elements. Notice that the decimation factor along the time or the space dimension may be different than two.

5.2 Recursive filter implementation

The implementation described in the previous section requires \( n \) processing elements (\( n \) being the number of lines in the image). The implementation proposed in the present section assumes that the image is read one pixel at a time, columns first, and requires \( O(\log(n)) \) processing elements. Each level is implemented in a separate processing element (pe) which computes the three high pass filter values (HP), and the low pass filter value (LP) which is passed on to the next processing element. To avoid several reads of the image, every pe has its own shift memory (cf. figure 6).

The first processing element is implementing the first level, which is described in figure 3. The pixels are read columns first, and thus three pixels are read and the corresponding partial results are stored in the shift memory. The next pixel read is processed with the partial results stored in the shift memory during the processing of the previous column. This algorithm is implemented in the following loop:

1. Read the next pixel from input into variable A
2. Read the next pixel from input into variable B
3. \( A = A + B \)
4. Read the next pixel from input into variable B
5. \( A = A + B \)
Write A on the shift memory M and shift M right.
Read the next pixel from shift memory M into variable A.
Read the next pixel from input into variable B.
A = A + B.
Write A on the LP output.

A similar loop is executed for the computation of the three high pass (HP) values. Notice that the shift memory is shifting right one position for every four pixels read. Thus this memory contains \( n/4 \) cells, \( n \) being the number of columns in the image. All the even processing elements execute the same loop. The program for the odd numbered processing elements is somewhat more complex because the values used to compute the HP and LP values are in four different columns (cf. figure 4). Notice that the \( pe1 \) works four times slower than the \( pe0 \). An alternative design would include only two processing elements working at the same speed and one shift memory per level, with multiplexers to address the proper shift memory.
6 Conclusion

In this paper, we presented a generalization of the Mallat pyramid for nD wavelets. We introduced a transformation to uniformize the equations defining the successive levels of the pyramid. The synthesis of wavelet functions allows us to explore the possible 2D Haar wavelet transforms and to study their properties. We proposed a methodology for implementing these wavelet transforms in parallel architectures like systolic arrays. More specifically, we have seen that there is a perfect match between the wavelet algorithms and the Multi-Phase multiRate Array (MPRA) architectures.

We are currently exploring various possible Haar wavelet transforms, and their implementation in an MPRA. We are studying their use in applications like edge detection and image compression.

References


AN A-R ALGORITHM FOR PATTERN RECOGNITION

D. B. Hoang and L. Esteban
Basser Department of Computer Science
The University of Sydney, NSW Australia 2006
doan@cs.usyd.edu.au

ABSTRACT

A novel learning and recognition algorithm is proposed in the paper. The algorithm is distinguished from existing algorithms [1] in two aspects: 1) the algorithm employs a Cauchy discrimination function which does not constrain possible locations of class regions; 2) the decision function measures not only the attraction force of the test pattern towards a class (a measure of similarity) but also the repulsion forces (a measure of unrelatedness) against other classes. The algorithm uses a distance metric normalised by a variation function. The variation function allows classes to have multiple regions in feature space. The algorithm provides control over the recognition process by allowing specification of negative samples, exclusiveness of classes. Initial results are encouraging and indicate the algorithm’s reliability.

1. Introduction

Machine Learning refers to the process of automatic generalisation using a recognition algorithm. Because of the automatic nature of machine learning, the description of input patterns and their categories must be systematically represented. The input patterns are generally represented by an array of attributes called feature vector, and the categories are represented by labels called classes. Feature vector is the link between an intelligent module and the outside world. Since the intelligence module is required to make conclusions about the outside world, the information conveyed by the features must be maximised. If poor features are used, the intelligence module is less likely to make accurate conclusions. Features thus provide the information necessary to distinguish input vectors. A recognition algorithm is, however, essential to actually make the decision concerning classes. A recognition algorithm must meet a number of requirements to ensure reliability. Apart from the internal mechanics for processing samples, a pattern recognition algorithm has to deal with various external issues that can affect its performance. These issues might include error samples, risk from making wrong decision, negative samples, variations in the feature dimensions, and missing features, etc.

The recognition algorithm proposed in this paper has been developed with considerations to these issues. The basic algorithm calculates the probability for each class that the class is the class of the unlabelled feature vector and returns the class with the greatest probability. The algorithm is distinguished from existing algorithms in that in two aspects. The algorithm employs a Cauchy discrimination function which allows for the incomplete nature of learning and does not constrain possible locations of class regions. The feature vector and the classes interact actively in the sense that the decision function measures not only the attraction force of the test pattern towards a class (a measure of similarity) but also the repulsion forces (a measure of unrelatedness) to other classes.

The paper is organised as follows. Section 2 defines the distance metric in terms of features of a sample. The feature vector is employed in the Cauchy discrimination function and in the evaluation of decision functions. Section 3 presents the algorithm, and section 4 presents some initial experimental results.
2. The Distance Metric

Most metric recognition algorithms use a decision function to determine which class a feature vector belongs. The class that produces the largest value for the function is the selected class.

Let \( x = (x[f_1], x[f_2], ..., x[f_n]) \) represent the feature vector where \( x[f_i] \) is the value for feature \( f_i \) of vector \( x \) and \( n = |F| \) is the number of features, where \( F = \{f_1, f_2, ..., f_n\} \) represents the feature set. Let \( x \) also represent the x. Let \( c \in C \) represent the recognised class that satisfies the decision function (Eqn.1), where \( C = \{c_1, c_2, ..., c_m\} \) represents the set of possible classes.

\[
d(x, c) > d(x, c') \quad \forall c \in C - \{c_1\} \quad (1)
\]

Let \( X_c = (x_1, x_2, ..., x_i) \) represent the training samples belonging to class \( c \). The classic decision algorithm is the Nearest Neighbour algorithm which uses Euclidean distance \( \Delta(x_1, x_2) \) as the distance between two sample points \( x_1, x_2 \), for the decision function \( d \) (Eqn.2, it is negative distance because the decision function uses greater than).

\[
\Delta^2(x_1, x_2) = \sum_{f \in F} (x_1[f] - x_2[f])^2
\]

\[
d(x, c) = -\min_{x_1 \in X_c} \sqrt{\Delta^2(x, x_1)} \quad (2)
\]

The nearest neighbour algorithm has three main deficiencies: 1) it does not return a class reliably when the closest distance is significantly large, the idea of closest neighbour is no longer meaningful, samples are sparse and unrelated; 2) it bases its decision on a single data point - the closest point. In many cases the measure of closeness to a group of samples is more powerful than closeness to just a single sample point; and 3) it cannot return an unknown class.

A more reliable decision could be made if the total influence of many points could be weighted. The k-nearest neighbour algorithm [2, 3] goes some way towards this by determining the most frequent class among the closest k points to the point in the feature space with the unknown class. In the next section we explore this idea further using the same distance metric.

3. The algorithm

To take into account the influence of many samples simultaneously over the test sample we define a generalisation function that represents the declining influence that a sample in feature space has on the surrounding region of feature space [4]. The function will take a form of a conditional probability, \( p \). This conditional probability is the probability that \( x_1 \) is distance \( \Delta \) from \( x_2 \) where \( x_2 \) is the centroid sample of a class \( c \).

\[
p(x_1, x_2, c) = \text{Prob} (x_1 \text{ is at a distance } \Delta \text{ from } x_2 | \text{the centre of class } c \text{ is at } x_2)
\]

(3)

In this paper we assume that the probability \( p \) only depends on the distance between the points in feature space. It means that the event \( x_1, x_2, c \) is independent of the event \( x_1, x_2, c \). Of course, the assumption can be relaxed if we know the probability structure of the sample space. The probabilities for all trained points are combined to give the final probability.

Basically the algorithm works as follows. Assuming that after the learning process, three classes have been identified, their samples and tentative boundaries are shown in figure 1a. We are given a unseen sample \( x \) and we want to determine the class of \( x \).

The algorithm first calculates three decision functions: the probability that \( x \) belongs to \( c_1 \) and \( x \) does not belong to \( c_2 \) or \( c_3 \); the probability that \( x \) belongs to \( c_2 \) and \( x \) does not belong to \( c_1 \) or \( c_3 \); the probability that \( x \) belongs to \( c_3 \) and \( x \) does not belong to \( c_1 \) or \( c_2 \); The maximum of these decision functions associates \( x \) with its most likely class. The algorithm can also return the indication that \( x \) may belong to a new class since \( x \) does not seem to fit in any of the existing classes. The new class can be established which \( x \) can be considered as a trained example.

The algorithm necessitates a discrimination function for measuring the influence a sample has on its surrounding; the decision function combines two effects: when \( x \) assumes the same class as a trained sample the measure of similarity resembles an attraction force; when \( x \) does not assumes the same class as the trained
sample, the measure of unrelatedness resembles an repulsive force; the discrimination is of a competitive manner; boundaries between classes are refined as more samples are included.

In the next section, we consider the normalisation of the distance function with respect to the variations of features, and we define the conditional probability function.

### 3.1 Feature Space Normalisation

Feature space variations are generally not equal in magnitude for each dimension [5]. To prevent excessive bias on the dimension where the magnitude and the variation of the feature are large, we need to normalise the Euclidean distance function. Variation in features depends on the feature and its class. For ease of computation, we suggest the following variance normalisation function which calculates the average difference between consecutive values of the given feature for all samples of the given class (More complicated normalisation function could also be employed).

\[
\nu(c, f) = \text{mean} \left[ \{|x_{i,c}[f] - x_i[f]|: x_i[f] \text{ are sorted}\} \right] = \frac{\max(x_i[f]) - \min(x_i[f])}{|X_c| - 1} \forall x_i \in X_c
\]

where \(|X_c|\) is the number of training samples belonging to class \(c\). The normalised Euclidean distance function becomes equation 5 where both points are assumed to belong to class \(c\).

\[
\Delta^2(x_1, x_2, c) = \sum_{f \in F} \left( \frac{x_1[f] - x_2[f]}{\nu(c, f)} \right)^2
\]

### 3.2 Probability Function Definition

The probability function can now be defined based on the normalised Euclidean distance. In this paper we employ a probability function based on the Cauchy probability density function [5].

\[
\text{Cauchy pdf} (x, \mu, \alpha) = \frac{1}{\mu \alpha} \frac{1}{1 + \left( x - \mu \right)^2} \alpha
\]

\[
= \frac{1}{\mu \text{scale}} \frac{1}{1 + \left( \text{distance - location} \right)^2} \text{scale}
\]

![Figure 1: Calculating the decision functions](image)

![Figure 2: Cauchy pdf with location = 0 and scale = 2](image)

The Cauchy pdf has a number of interesting properties. It is symmetrical about the location \(\mu\), and a scale factor \(\alpha\) (figure 1). The mean and standard deviation of the distribution are indeterminate. This factor is quite important for a learning and recognition system since we assume that the generalisation is never complete, there may be some samples we have not seen yet. This means that the function does not constrain possible samples and possible locations of class regions. In this paper we use the Cauchy pdf as a form of discrimination function which takes into account the influence of a trained sample over an unseen sample based on the distance between them. The trained sample is assumed at the location \(\mu\). We assume the scale to be 1 (\(\alpha = 1\)). The probability function we employ for our purpose here is then the probability that two points belonging to the same class, and is given by:

\[
p(x_1, x_2, c) = \frac{1}{1 + \Delta^2(x_1, x_2, c)}
\]

Suppose another sample \(x_i\) also belongs to class \(c\), the probability that \(x_i\) is distance \(\Delta^2(x_1, x_i, c)\) is given by

\[
p(x_1, x_i, c) = \frac{1}{1 + \Delta^2(x_1, x_i, c)}
\]
If we only have two samples which belong to class c. The probability of x belongs to class c is then

\[
p((x_1, x_2, c) \cup (x_1, x_1, c)) = p(x_1, x_2, c) \land (x_1, x_3, c) \\
= 1 - p((x_1, x_2, c) \land (x_1, x_3, c)) \\
= 1 - [(1 - p(x_1, x_2, c))(1 - p(x_1, x_3, c))] \tag{9}
\]

In general, if we are given all samples, i's, of class c then the probability that an unknown vector x belonging to class c is given by

\[
p_{\text{samples}}(x, c) = 1 - \prod_{i \in X_c}(1 - p_{\text{sample}}(x, i, c)) \tag{10}
\]

Equation 10, however, does not take into account the influence of samples which are known to belong to other classes, and samples which are known not to belong to class c (negative samples). Taking these samples into account, the final probability that x belongs to class c, given all samples is given by

\[
p(x \text{ belongs to c given all samples of class c} \\
p \text{ and x does not belong to any other class} \\
given their respective samples of that class} \tag{11}
\]

\[
= p_{\text{samples}}(x, c) \prod_{i \in N_c \cup \bar{c}}(1 - p_{\text{samples}}(x, i))
\]

where \(N_c\) represents the set of classes that exclude class c, and \(\bar{c}\) represents the negative class for class c (the negative class can be returned if no positive class matches)

The decision function for x belonging to class c is then

\[
d(x, c) = p_{\text{class}}(x, c) \\
= p_{\text{samples}}(x, c) \prod_{i \in N_c \cup \bar{c}}(1 - p_{\text{samples}}(x, i)) \tag{12}
\]

The probability of belonging to a unknown class is given by

\[
p_{\text{unknown class}}(x) = \prod_{c \in C}(1 - p_{\text{class}}(x, c)) \tag{13}
\]

The algorithm returns the class with maximum probability, the probability of matching as well as the probability that x belonging to a unknown class.

4. Experiments on Artificial Data Sets

This section presents some results of experiments with the basic recognition algorithm. An artificial data set was generated by a feature space oracle. The oracles were designed mindful of the fact that class regions that have no overlapping ranges for every feature are easy to recognise. Therefore, a data set was used where the classes form linked toruses in feature space because there is overlap in every feature. The data set has the two toroidal class regions touching. Each torus represents one class. This should increase the number of recognition errors because test samples near the interface will fall approximately midway between the class regions defined by the training samples. The data set was produced by generating random points contained within the torus shapes with an even distribution. The data set consists of 1000 samples of class A and 1000 samples of class B as shown in figure 3

The features used for the experiment were three Cartesian coordinates of the samples. The experiments were performed as a series of repeated tests where the number of samples in the training set was increased by 10 after each test was tested 10 times. The experiment started with 10 training samples. The tests were continued until the training set had 150 samples. Training set samples were randomly selected from the whole data set. No effort was made to keep the number of samples for both classes the same.

The results can be seen in the graph below. The graph measures the number of samples used for training versus the number of correctly recognised samples divided by the number of test samples averaged over the number of repetitions. The graph also include error bars indicating the maximum and minimum recognition rate encountered during the repetitions.

Initial results show (figure 4) that the basic algorithm

- has a fast learning curve. It only requires a small number of samples before it achieves a reasonable recognition rate,
Figure 3: The two toruses. a) feature space; b) samples
• performs consistently for a given training set size. This means that the minimum recognition rate is closer to the average rate indicating that the algorithm is less sensitive to the training samples chosen.

5. Conclusion

A novel recognition algorithm has been developed. The algorithm can return the class of a unlabelled sample as well as the probability of the sample belonging to a unknown class. The algorithm employs a Cauchy discrimination function in combination with a normalised-by-variation distance metric function. This combination allows classes to have multiple regions in feature space. The decision function employed in the algorithm measures not only the attraction force of the test pattern towards a class but also the repulsion forces against other classes. This makes it possible to extract test samples that lie in the overlapping regions of two or more classes. The algorithm also provides control over the recognition process by allowing specification of negative samples, exclusiveness of classes. Initial results are encouraging and indicate the algorithm's reliability. Experiments will be performed to compare the results by the algorithm with those obtained from existing methods.

REFERENCES

1. Lippmann, R. P., “Pattern Classification Using Neural Networks” IEEE Communications Magazine, vol. 27, no. 11, pp. 47-64.


"True Colour" Auto-Stereograms

Allan Bromley and Barbara Karakassidis
Computer Science, University of Sydney

Abstract

Auto-stereograms combine a pair of stereo images into a single printed image that can be viewed divergently without any special optical apparatus. They have achieved world wide popularity as puzzles in Magic Eye 3-D Images. The technique can be extended to provide much more faithful renderings of 3-D objects. In this paper we show how "true colours" can be applied to objects.

Stereo Vision

Stereo vision, 3-D or "solid" vision, is possible because the left and right eyes view a scene from slightly different vantage points. There are thus small disparities between the images received on the two retinas. Features extracted from the two images are sent from the retinas to the visual cortex of the brain where corresponding points from the two eyes are brought to adjacent neural columns. The brain then interprets the small disparities between the images as a 3-D image in depth.

Artificial stereo drawings were first produced by Wheatstone in 1832 and viewed by a mirror stereoscope of his devising. With the advent of photography, stereo images enjoyed great popularity at the end of the 19th century. A Brewster twin lens viewer was commonly employed. Stereo enjoyed a resurgence of popularity in the 1950s and the ViewMaster system of circular cards holding pairs of full colour transparencies became well known. A further resurgence of interest in stereo imaging now appears to be in progress. Stereo photogrammetry has great importance in large scale mapping. Unfortunately, stereo pairs are small in size, and the viewing apparatus is expensive.

Amongst other methods of presenting stereo images only the Anaglyph technique is widely used. A printed image is viewed though glasses with complementary colours, typically red and green, covering the eyes. Two images are over-printed in the same two complementary colours. When viewed through the green lens the green printed image appears white and is indistinguishable from the background. The red printing appears black, and the eye sees the red image as a single black and white image through the green lens. Conversely, the green printed image is seen as a black and white image through the red lens. The two eyes thus receive separate stereo images from the overprinted pair.

Large anaglyphs can be viewed with only simple optical apparatus and the technique has been demonstrated as an effective method for 3-D movies. [E.g. Hitchcock's "Dial M for Murder".] The technique is relatively cheap, but colour images are not possible with the basic anaglyph technique.

Random-dot stereograms were first demonstrated by Bela Julesz in 1959 [Thimbleby & Neesham, 1993]. Each image consists of the same field of random dots, but in one image part of the field is
displaced a small amount. When viewed as a stereoscopic pair the displaced field is seen raised and floating above the background. This demonstrates that the features extracted by the visual system and used to give the depth impression need not be "natural" features of objects in the scene. Any detailed texture that can be correlated between the images in the two eyes seems sufficient to give a 3-D effect.

Auto-stereograms, in which a stereo image can be viewed without special apparatus, were first developed by Tyler in 1979. [Rheingold, 1994] The method is based on the "wallpaper illusion" discovered by Brewster. Two strips in a narrow repeat wallpaper print may be brought into coincidence by diverging the lines of sight of the two eyes to a point behind the wallpaper - looking "through" the wallpaper or "staring into the distance". Two separately printed images from adjacent strips of the wallpaper are fused to form a single image. Any disparity between the prints is seen as a startling 3-D effect.

In the "Magic-Eye" auto-stereograms, pixels in the wallpaper strips (which might be a random dot pattern) are displaced from strip to strip to give a 3-D image of a desired object. Viewing these images is not particularly easy as the convergence of the lines of sight of the two eyes and the accommodation of the lenses of the eyes must be controlled separately. (These are normally coordinated with one another and act as auxiliary cues in normal depth perception.) The technique can be learned, but is probably acquired more easily by children than adults. Smart marketing has transformed the difficulty of viewing these images into a "puzzle" to challenge viewers.

Although colours are commonly employed in random dot auto-stereograms these only form part of the "texture" giving the 3-D effect. A black and white photocopy of a Magic-Eye image will give the same 3-D effect. The colours are certainly not applied as "true colours" to the objects seen in the 3-D image. The effect might be of a paisley painted shark floating in a paisley painted sea. The wallpaper effect is usually very strong, with very obvious repeating patterns, and the degree of reality achieved is generally poor. Although some images are quite smartly designed, the majority are unlikely to have an enduring public interest.

Our purpose has been to develop the random dot auto-stereogram technique to the point where it can be used as an effective medium for publication of 3-D images requiring no special viewing apparatus. We seek to provide a full range of colour and tonal effects in the perceived objects. In this paper we demonstrate a technique by which effective "true colour" can be applied to objects in an auto-stereogram.

Making Auto-Stereograms

The horizontal lines in an auto-stereogram are independent of one another. All of the geometry of the 3-D depth perception can be described in a plane containing the viewer's eyes and a horizontal line of the image. The vertical extent of the objects provides, of course, coherence from line to line. The auto-stereogram rendering is inherently a raster scan process.

Figure 1 shows the basic geometry of auto-stereogram depth perception with divergent viewing. Small patterns are repeated across the image giving the basic
wallpaper effect. The repeated patterns can be thought of as image pixels.

The separation of the wallpaper repeats determines the depth perception. A slightly closer spacing of the patterns brings the perceived object forward towards the viewer as shown in the center of the figure. Since the reduction in spacing can, in practice, only be in units of the pixel size the image resolution controls the depth resolution. With practical geometry the depth resolution is 4-5 times more coarse than the x-y pixel resolution. Unless the x-y resolution is very fine, the poor z resolution may cause the image to appear to be constructed of a series of cutouts on parallel planes.

Given a description of the scene as a depth, z(x,y), for each pixel in the image plane it is straightforward to link those pixels which combine to form points on the perceived object. Our algorithm, though developed independently, is similar to that of Tyler and Clarke (1979) [Thimbleby & Neesham, 1993]. If all linked pixels are rendered the same (colour and grey scale) a conventional auto-stereogram results. Random dots are sufficient, but a more "artistic" choice is made in most commercial products.

A simple file of z(x,y) does not convey enough information to correctly render all points that are visible to one eye but not the other. In practice the defects are minor and of little consequence at this stage of development.

True-Colour Auto-Stereograms

The perception of an object point, O, is formed from the information from one point, L, in the auto-stereogram viewed by the left eye, and another point, R, viewed by the right eye. We suppose that the colour perceived in the object is simply a function of the colours of the points viewed by the left and right eyes:

\[
Col(O) = fn[Col(L), Col(R)]
\]

In standard colour theory it is known that the combination of colours is a linear process. Further, it is known that the Red, Green, Blue phosphors used in computer terminals forms a basis set for the linear colour space.

The standard colour theory is essentially a theory of monocular vision, and some or all of the colour combination may take place on the retina. We assume, however, that a similar model underlies binocular colour vision, at least as a first approximation. Our object colour is thus assumed to be formed of three functions of the form:

\[
Blue(O) = \frac{1}{2}[Blue(L) + Blue(R)]
\]

Figures 1 and 2 show the application of this idea. Suppose that the object is coloured blue, B, and that the background is a rather neutral shade, BG. If the left and right eye points are both coloured B then the object point also has the colour B, as shown in the center of the figures. If we move right then we wish to perceive a background point coloured BG but the left eye point is already coloured B. We must therefore colour the right eye point a red shade, R, so that:

\[
BG = \frac{1}{2}[B + R]
\]

As we move further right we find the next linked point must be coloured B, the next R, and so on.

The occurrence of phantom images of the object, to left and right of the principal image, coloured alternately in colours complementary with respect to the
background, is characteristic of our technique. At the extreme left and right, beyond the area of stereo perception, these phantom images cannot be eliminated.

Consideration of Figure 1 shows that the colour of $n$ perceived object points is determined by the colouring of $n+1$ points in the auto-stereogram. We thus have one degree of freedom in the colour rendering. Once the colour of one point is fixed the colour of the remaining $n$ is then determined. At present we commence the rendering in the center of the auto-stereogram, rendering the object in its true colour. Figures 3 and 4 show geometric objects rendered using our technique.

**Colour Combination**

We have found that when colour differences between linked points in the auto-stereogram is small they do combine according to the linear process assumed above.

When colour differences are large we do not get colour combination but instead have perceptual rivalry. [Levine & Shefner, 1991] A pattern coloured yellow for one eye and blue for the other, for example, are so different that the images contend with one another. Sometimes it is seen as blue, at other times as yellow, but not as a combined (green?) colour. The visual cortex appears to be attempting to find a consistent interpretation but fails with one eye sometimes dominating, and sometimes the other. If the patterns are extended in size the perceived image seems to swim backwards and forwards between the two alternative colour interpretations in moving spatial patches.

Despite the failure of colour combination in perceptual rivalry, stereo interpretation of the object seems largely unimpaired. In the extreme case, a black and white image in one eye may be combined with its negative in the other to give a 3-D stereo image but with an "iridescent" shading. [Judge, 1926]

Suppose the RGB colour components are each measured on a scale from 0.00 to 1.00. We have found that differences in the red component up to 0.25 for adjacent linked pixels are readily merged. For blue and green the limits are tighter: up to 0.15 and 0.10 respectively. There appears to be some difference between individuals and some accommodation as one continues to view an image. We have not had an opportunity to attempt to define the psychophysical limits more carefully nor to explore the implications our results have for the theory of visual perception.

It is clear that the one degree of freedom that exists in colouring a set of linked pixels in the auto-stereogram should be used to minimise the maximum colour contrast that exists between adjacent linked pixels. An implementation of this idea is in progress. We expect then to be able to exploit a richer colour palate.

**References**


