Fitting and tracking of a scene model in very low bit rate video coding

Paul Antoszczyszyn

A thesis submitted for the degree of Doctor of Philosophy
The University of Edinburgh
April 1998
Abstract

In the contemporary world communication technology has an immense influence on the way we work and behave. For many years now the telephone has been the most commonly used means of interactive communication. Recently, due to the standardisation and commercial application of moving image compression techniques (MPEG, MPEG-II, H.263, H.263+) and the ever increasing power of personal computers, the interest in interactive video communication (videophone) applications has grown considerably.

Most of the research concerning video compression techniques avoids the topic of extremely low bit rates (required e.g. for mobile communication). There is currently no technique dedicated to encoding the video for such low data rates (below 10 kbit/s). In most cases a reduction in frame-rate and heavy quantisation would be applied to an existing algorithm designed for a higher data rate. The resulting artefacts would in many cases prevent the recognition of the speaker in a head-and-shoulders scene.

In recent years, due to the development of the MPEG-IV standard, there has been a growing interest in model based video coding techniques with algorithms utilising semantic knowledge (wire-frame models) about the scene offering the highest compression ratio.

This thesis describes an investigation into the topic of semantic model based coding of typical videophone scenes (head-and-shoulders and head-only). New techniques for automatic fitting of the semantic wire-frame are described and tested. Finally a new algorithm for automatic tracking and a unified approach to both fitting and tracking are presented.

Due to very encouraging feedback from other researchers working in the same area, it was possible to publish the results of investigations described in this thesis in 15 journal and conference papers. These are listed at the end of this thesis and one of the published papers is included.
Declaration of originality

I hereby declare that this thesis and the work reported herein was composed and originated entirely by myself, in the Department of Electrical Engineering at the University of Edinburgh.

Paul Antosczyszyn
I would like to thank Dr John Hannah and Professor Peter Grant, my supervisors, for their continuous support, invaluable advice and guidance during the course of this PhD, and for reading this thesis.
# Contents

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>List of figures</td>
<td>vii</td>
</tr>
<tr>
<td>List of tables</td>
<td>xiv</td>
</tr>
<tr>
<td>Abbreviations</td>
<td>xvi</td>
</tr>
<tr>
<td><strong>1. Introduction</strong></td>
<td>1</td>
</tr>
<tr>
<td>1.1 Theme of the thesis</td>
<td>1</td>
</tr>
<tr>
<td>1.2 Motivation</td>
<td>2</td>
</tr>
<tr>
<td>1.3 Thesis organisation</td>
<td>3</td>
</tr>
<tr>
<td><strong>2. Moving image compression systems</strong></td>
<td>5</td>
</tr>
<tr>
<td>2.1 Introduction</td>
<td>5</td>
</tr>
<tr>
<td>2.2 Existing moving image compression techniques</td>
<td>6</td>
</tr>
<tr>
<td>2.2.1 Waveform based approach</td>
<td>6</td>
</tr>
<tr>
<td>2.2.1.1 Block DCT transform based coding</td>
<td>7</td>
</tr>
<tr>
<td>2.2.1.2 Vector quantisation based coding</td>
<td>12</td>
</tr>
<tr>
<td>2.2.1.3 Fractal based coding</td>
<td>15</td>
</tr>
<tr>
<td>2.2.1.4 Wavelet based coding</td>
<td>24</td>
</tr>
<tr>
<td>2.2.2 Model based approach</td>
<td>27</td>
</tr>
<tr>
<td>2.2.2.1 Object oriented coding</td>
<td>27</td>
</tr>
<tr>
<td>2.2.2.2 Semantic model or knowledge based coding</td>
<td>31</td>
</tr>
<tr>
<td>2.2.3 MPEG-4 and layered coders</td>
<td>35</td>
</tr>
<tr>
<td>2.3 Summary</td>
<td>44</td>
</tr>
</tbody>
</table>
3. Knowledge based coding and facial recognition

3.1 Introduction

3.2 Main problems of knowledge based coding

3.2.1 Wire-frame fitting (adaptation)

3.2.2 Wire-frame tracking

3.2.3 Other problems: images and speech

3.2.4 Summary

3.3 The use of face recognition techniques

3.3.1 Feature extraction algorithms

3.3.2 Facial recognition techniques

3.3.2.1 Artificial neural networks based systems

3.3.2.2 Principal components analysis based systems

3.3.3 Summary

3.4 Conclusion

4. Tracking the subject in the scene

4.1 Introduction

4.2 The tracking algorithm

4.2.1 System preparation

4.2.2 Eigenvalue decomposition of a sequence of images

4.2.3 Automatic tracking

4.2.4 Best match measure and upgrading of initial set

4.2.4.1 Distance measure

4.2.4.2 Upgrading of the initial set

4.2.4.3 Testing method

4.2.4.4 Reducing the dimensionality of the principal component space

4.2.5 Results of tests on head-and-shoulders video scenes

4.3 Local motion tracking
5. Fitting of a generic wire-frame model to the scene

5.1 Introduction

5.2 Initial attempts: feature extraction

5.2.1 Lips centre extraction from motion

5.2.2 Segmentation and mathematical morphology

5.3 Application of facial data-bases

5.3.1 Introduction

5.3.2 Correlation based algorithm

5.3.2.1 Facial data-base pre-processing

5.3.2.2 Processing of the unknown image

5.3.2.3 Results of the tests

5.3.2.4 An improved algorithm

5.3.2.5 Results of tests with the improved algorithm

5.3.2.6 Summary

5.3.3 Principal component analysis based algorithm

5.3.3.1 Facial data-base preparation

5.3.3.2 On-line algorithm

5.3.3.3 Results of the tests and summary

5.4 Summary

6. Fitting and tracking: a unified approach

6.1 Introduction

6.2 Algorithm description

6.2.1 Generic data-base preparation

6.2.1.1 Spatial equalisation
6.2.1.2 Greyscale equalisation (data-base preparation)  
6.2.2 On line processing of a video sequence  
\hspace{1em} 6.2.2.1 Initial tracking  
\hspace{1em} 6.2.2.2 Greyscale equalisation (standard tracking)  
\hspace{1em} 6.2.2.3 Standard tracking  
6.3 Results of the tests  
6.4 Summary

7. Conclusions and future research  
\hspace{1em} 7.1 Introduction  
\hspace{1em} 7.2 Summary of practical achievements  
\hspace{1em} 7.3 Future work  
\hspace{1em} 7.4 Model based approach to moving image coding: final remarks

References

Appendix 1: Code example: histogram equalisation

Appendix 2: Published articles
List of figures

2.1 Moving image coding techniques 6
2.2 Frame dependency in MPEG coding 8
2.3 'Zig-zag' scan 9
2.4 MPEG Encoder 9
2.5 MPEG Decoder 10
2.6 Binary split in TSVQ 13
2.7 Fractal image compression, mathematical foundations and implementation 16
2.8 Fractal analysis of an image 17
2.9 Domain blocks classification 18
2.10 R-frames 21
2.11 Fractal video coding 21
2.12 A novel spatio-temporal split 22
2.13 Heuristic segmentation of an image 24
2.14 Motion compensated wavelet based moving image encoder 26
2.15 Principle of wavelet decomposition 26
2.16 Encoder in the OBASC system 28
2.17 Scene decomposition in OBASC 30
2.18 Candide wire-frame 32
2.19 Candide -2 wire-frame 32
2.20 Eyes' and lips' wire-frame model details 33
2.21 Knowledge based moving image coding system 34
2.22 Layered video coder 36
2.23 MPEG-4 tools, algorithms and profiles 39
2.24 MPEG-4 system overview 40
2.25 Typical 'head-and-shoulders' scene subdivided into two independent VOPs 42
2.26 Standard and contour macroblocks in MPEG-4 42
2.27 Padding VOP 43
3.1 Knowledge based coding and face recognition for analysis of a videophone scene
3.2 'Rough fit' (Aizawa)
3.3 'Accurate fit' (Aizawa)
3.4 Changing action units (original model: top left corner)
3.5 Mathematical morphology based edge detector
3.6 Simplified *Candide*
3.7 Original (left) and mosaic images (middle, right)
3.8 Model of the human face (left) and individual features the system searches
3.9 String model of a face
3.10 String model of a face
3.11 Extraction points: each vertex denotes an extraction point
3.12 Processing of the sub-image containing lips
3.13 A multi-layer perceptron artificial neural network, each circle represents an artificial neurone, the arrows point in the direction of the signal
3.14 MLP applied to character recognition
3.15 Analysis of an image using artificial neural networks
3.16 Pre-processing of 20×20 window (top to bottom): mask eliminating background pixels, original images, line function images, lightning corrected images, histogram equalised images.
3.17 Test images (top), PCA-based recognition (bottom)
4.1 Tracking of important facial features
4.2 The left eye, the right eye, the nose and the lips sub-sequences
4.3 Extraction reference points for the sub-images of the eyes, the lips and the nose
4.4 Scanning sub-images to obtain 1D vectors
4.5 Normalisation of the sub-sequence of the eyes
4.6 Eigen-images of differential images presented in Figure 4.5
4.7 Processing of the current frame
4.8 Initial set and extracted set
4.9 Dynamic upgrade of the initial set
4.10 Testing error measures on sub-image sequences extracted from *Miss America*
4.11 Dynamically updated initial set, distance A, frame 45 and 48: eye tracking
4.12 Dynamically updated initial set, distance A, frame 61 and 93: lip tracking
4.13 Dynamically updated initial set, distance B, frame 45 and 48: eye tracking
4.14 Dynamically updated initial set, distance B: frame 100 and 110: lip tracking
4.15 Measure A (left) and measure B (right) tracking test stills: frame 22
4.16 Measure A (left) and measure B (right) tracking test stills: frame 62
4.17 Measure A (left) and measure B (right) tracking test stills: frame 124
4.18 Measure A (left) and measure B (right) tracking test stills: frame 136
4.19 Error measured as the Euclidean distance in image space
4.20 Measure A (left) and measure B (right) left eye tracking error profiles
4.21 Measure A (left) and measure B (right) right eye tracking error profiles
4.22 Measure A (left) and measure B (right) nose tracking error profiles
4.23 Measure A (left) and measure B (right) lips tracking error profiles
4.24 Measure B describes the distance from a set of images rather than an individual image
4.25 Eigenvector images arranged in order of decreasing ‘importance’
4.26 Claire initial sets for the eyes, the nose and the lips
4.27 Claire difference images for the eyes, the nose and the lips
4.28 Claire eigenvector images for the eyes, the nose and the lips
4.29 Trevor initial sets for the eyes, the nose and the lips
4.30 Trevor differential images for the eyes, the nose and the lips
4.31 Trevor eigenvector images for the eyes, the nose and the lips
4.32 Manual extraction of features from every fifth frame only
4.33 Miss America: The left eye (left) and the right eye (right) tracking error profiles
4.34 Miss America: The nose (left) and the lips (right) tracking error profiles
4.35 Claire: The left eye (left) and the right eye (right) tracking error profiles
4.36 Claire: The nose (left) and the lips (right) tracking error profiles
4.37 Car phone: The left eye (left) and the right eye (right) tracking error profiles
4.38 Car Phone: The nose (left) and the lips (right) tracking error profiles
4.39 Grandma: The left eye (left) and the right eye (right) tracking error profiles
4.40 Grandma: The nose (left) and the lips (right) tracking error profiles
4.41 Salesman: The left eye (left) and the right eye (right) tracking error profiles
4.42 Salesman: The nose (left) and the lips (right) tracking error profiles 98
4.43 The left eye (left) and the right eye (right) tracking error profiles 98
4.44 Trevor: The nose (left) and the lips (right) tracking error profiles 98
4.45 Tracking Miss America; frames: 0, 20 100
4.46 Tracking Miss America; frames: 85, 98 100
4.47 Tracking Miss America; frames: 110, 120 100
4.48 Tracking Miss America; frames: 124, 140 100
4.49 Tracking Claire; frames: 0, 75 101
4.50 Tracking Claire; frames: 95, 105 101
4.51 Tracking Claire; frames: 140, 145 101
4.52 Tracking Claire; frames: 155, 167 102
4.53 Tracking Grandma; frames: 30, 40, 70 102
4.54 Tracking Grandma; frames: 120, 180 102
4.55 Tracking Grandma; frames: 340, 410, 440 102
4.56 Tracking Grandma; frames: 500, 600, 700 102
4.57 Tracking Car Phone; frames: 30, 55, 75 103
4.58 Tracking Car Phone; frames: 80, 105, 170 103
4.59 Tracking Car Phone; frames: 200, 290, 320 103
4.60 Tracking Car Phone; frames: 333, 377, 399 103
4.61 Tracking Salesman; frames: 0, 50 103
4.62 Tracking Salesman; frames: 80, 120 104
4.63 Tracking Salesman; frames: 140, 210 104
4.64 Tracking Salesman; frames: 340, 351 104
4.65 Tracking Trevor; frames: 0, 20 105
4.66 Tracking Trevor; frames: 30, 40 105
4.67 Tracking Trevor; frames: 50, 70 105
4.68 Tracking Trevor; frames: 80, 99 105
4.69 Sub-images centred on the vertices describing facial features 107
4.70 Miss America initial sets for the left and right eye corners 107
4.71 Miss America differential images for the left and right eye corners 108
4.72 Miss America eigenvector images for the left and right eye corners 108
4.73 Miss America initial sets for the lips 108
4.74 Miss America differential images for the lips 108
4.75 Miss America eigenvector images for the lips 108
4.76 Miss America; frames 0 and 21 local motion tracking 109
4.77 Miss America; frames 32 and 46 local motion tracking 109
4.78 Miss America; frames 90 and 97 local motion tracking 109
5.1 Subsampling the area occupied by the lips 112
5.2 Macropixel 113
5.3 Macropixel search area 114
5.4 Frame 16 of Miss America sequence (left) with superimposed motion field (right) 114
5.5 Frame 17 of Miss America sequence (left) with superimposed motion field (right) 114
5.6 Frame 18 of Miss America sequence (left) with superimposed motion field (right) 115
5.7 Frame 19 of Miss America sequence (left) with superimposed motion field (right) 115
5.8 Frame 20 of Miss America sequence (left) with superimposed motion field (right) 115
5.9 Frame 21 of Miss America sequence (left) with superimposed motion field (right) 115
5.10 Image segmentation approaches 116
5.11 Clockwise from the top left corner: Miss America (original), variance criterion, average criterion and dynamic criterion. 118
5.12 Clockwise from the top left corner: Miss America pre-processed with morphological opening followed by morphological closing, variance criterion, average criterion and dynamic criterion. 119
5.13 Top left: Claire - no pre-processing applied; top right: split-and merge applied to the image on the left; bottom left: pre-processing applied to the top left image, bottom right: split and merge applied to the bottom left image. 120
5.14 Split and merge technique localises lips at lower resolution (left) but fails to do so if the same algorithm is applied to the full resolution image (right) 121
5.15 Candide model consisting of 79 vertices and 108 triangles 122
5.16 Widow image (left) fitted manually with the Candide wire-frame (right) 123
5.17 Computer-generated expressions 123
5.18 Computer generated expression (left) and rotation (right) 123
5.19 Computer generated rotations 124
5.20 Example images from the MIT facial archives 124

xi
5.21 Manual fitting example
5.22 Testing manual fitting using specification from Table 5.2 (au0 to au3)
5.23 Testing manual fitting using specification from Table 5.2 (au4 to au7)
5.24 On-line processing of the unknown image
5.25 Vertices describing the left and the right eye
5.26 Vertices describing the lips and the nose
5.27 Successful (left) and unsuccessful (right) fitting
5.28 Manual fit (left) and successful automatic fit (right)
5.29 Testing automatic wire-frame fitting accuracy
5.30 Reconstruction artefacts due to misjudged fit in the corner of the lips
5.31 Dynamically increasing the number of sub-images containing lip shapes
5.32 The area of adjustment in the third (fine) stage of the algorithm
5.33 Testing automatic wire-frame fitting accuracy after application of the fine fitting step (action units and global motion as in Table 5.4)
5.34 An apparently successful fit (left) and a verified successful fit (right)
5.35 Corner of the lips fitting error
5.36 The left eye, the right eye, the nose and the lips sub-sequences
5.37 Extraction reference points for the sub-images of the eyes, the lips and the nose
5.38 PCA-based wire-frame fitting facial data-base preparation
5.39 On-line automatic wire-frame fitting using PCA approach
5.40 Testing the accuracy of automatic fitting using PCA-based method, action units as in Table 5.4
5.41 Testing the accuracy of automatic fitting using PCA-based method in the presence of facial occlusions (moustache, beard), action units as in Table 5.4
6.1 Framework of an automatic fitting/tracking algorithm
6.2 Imagery used to construct the facial data-base
6.3 Face extraction (Aizawa’s ‘rough fit’)
6.4 Facial features extraction (Aizawa’s ‘accurate fit’)
6.5 Extraction rules
6.6 Images before (left) and after (right) histogram equalisation
6.7 Global and local histogram equalisation
6.8 Pre-processing prior to application of the principal component analysis
6.9 Initial frames of the analysed sequence
6.10 Modification to the wire-frame fitting algorithm to accommodate equalisation

6.11 Usual shape of histogram (left) and its linearised version (right)

6.12 Discrete (8 levels, 64 x 64 pixels) histogram before (left) and after equalisation (right)

6.13 Initial tracking of Claire (frame 0) without (left) and with (right) equalisation

6.14 Initial tracking of Claire (frame 5) without (left) and with (right) equalisation

6.15 Initial tracking of Claire (frame 10) without (left) and with (right) equalisation

6.16 Initial tracking of Salesman (frame 0) without (left) and with (right) equalisation

6.17 Initial tracking of Salesman (frame 5) without (left) and with (right) equalisation

6.18 Initial tracking of Salesman (frame 10) without (left) and with (right) equalisation

6.19 Principal component data-bases are changed after S frames

6.20 Claire (frame 93) manual (left) and automatic (right) initialisation

6.21 Claire (frame 102) manual (left) and automatic (right) initialisation

6.22 Claire (frame 104) manual (left) and automatic (right) initialisation

6.23 Salesman (frame 202) manual (left) and automatic (right) initialisation

6.24 Salesman (frame 290) manual (left) and automatic (right) initialisation

6.25 Salesman (frame 300) manual (left) and automatic (right) initialisation

7.1 Semantic based tracking control

7.2 Model-waveform hybrid codec possible results

A.1 Original histogram $h(l)$, transformation function $T(l)$, and new histogram $h'(s)$
List of tables

2.1 Approximate data rates of an uncompressed video signal ............................................. 5
2.2 MPEG control levels ........................................................................................................ 11
2.3 Fractal block coding versus vector quantisation ............................................................. 19
2.4 Results of fractal compression applied to video sequences ........................................... 20
2.5 Requirements of MPEG-4 ............................................................................................... 38
3.1 Selected action units (AUs) ............................................................................................. 48
3.2 Selected phoneme classes ............................................................................................... 58
3.3 Phoneme vectors .............................................................................................................. 58
4.1 Mean tracking error and standard deviation: Miss America (150 frames) ...................... 90
4.2 Sequences tested ............................................................................................................. 94
4.3 Manual extraction sample .............................................................................................. 95
4.4 Tracking error results for Miss America and Claire ....................................................... 99
4.5 Tracking error results for Car Phone and Grandma ...................................................... 99
4.6 Tracking error results for Salesman and Trevor ............................................................ 99
4.7 The left and the right eye tracking quality ....................................................................... 106
4.8 The nose and the lips tracking quality .......................................................................... 106
5.1 Splitting stage results ................................................................................................... 120
5.2 Action units used for testing of the accuracy of fitting ................................................... 126
5.3 Isotropic templates for edge detector ............................................................................ 127
5.4 Action units and global motion vectors used for fitting accuracy testing ..................... 133
5.5 Lips' dynamic data-base action units ............................................................................ 135
5.6 Testing of the error measure .......................................................................................... 139
6.1 Tracking error results for Miss America and Claire: automatic initialisation ............... 166
6.2 Tracking error results for Car Phone and Grandma: automatic initialisation ............... 166
6.3 Tracking error results for Salesman and Trevor: automatic initialisation ..................... 166
6.4 The left and the right eye tracking quality: automatic initialisation ............................. 167
6.5 The nose and the lips tracking quality: automatic initialisation 167
6.6 Execution times for tested sequences 168
Abbreviations

<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ANN</td>
<td>Artificial Neural Network</td>
</tr>
<tr>
<td>AU</td>
<td>Action Unit</td>
</tr>
<tr>
<td>AV</td>
<td>Audio-Video</td>
</tr>
<tr>
<td>CBR</td>
<td>Constant Bit Rate</td>
</tr>
<tr>
<td>CIF</td>
<td>Common Interface Format: 352x288 (Y), 176x144 (UV)</td>
</tr>
<tr>
<td>DCT</td>
<td>Digital Cosine Transform</td>
</tr>
<tr>
<td>DFSVQ</td>
<td>Dynamic FSVQ</td>
</tr>
<tr>
<td>DMX</td>
<td>De-multiplexer</td>
</tr>
<tr>
<td>FACS</td>
<td>Facial Action Coding System</td>
</tr>
<tr>
<td>FSVQ</td>
<td>Finite state VQ</td>
</tr>
<tr>
<td>GOP</td>
<td>Group of Pictures</td>
</tr>
<tr>
<td>KBASC</td>
<td>Knowledge-Based Analysis-Synthesis Codec</td>
</tr>
<tr>
<td>LVQ</td>
<td>Lattice VQ</td>
</tr>
<tr>
<td>MAD</td>
<td>Mean Absolute Difference</td>
</tr>
<tr>
<td>MC-DCT</td>
<td>Motion Compensated Discrete Cosine Transform</td>
</tr>
<tr>
<td>MLP</td>
<td>Multi-Layer Perceptron</td>
</tr>
<tr>
<td>MPEG</td>
<td>Moving Picture Experts Group</td>
</tr>
<tr>
<td>MSD</td>
<td>Mean Square Difference</td>
</tr>
<tr>
<td>MSDL</td>
<td>MPEG Syntactic Description Language</td>
</tr>
<tr>
<td>MSVQ</td>
<td>Multi Stage Hierarchical VQ</td>
</tr>
<tr>
<td>OBASC</td>
<td>Object-Based Analysis-Synthesis Codec</td>
</tr>
<tr>
<td>PCA</td>
<td>Principal Component Analysis</td>
</tr>
<tr>
<td>PSTN</td>
<td>Public Switched Telephone Network</td>
</tr>
<tr>
<td>QCIF</td>
<td>Quarter Common Interface Format: 176x144 (Y), 88x72 (UV)</td>
</tr>
<tr>
<td>SVD</td>
<td>Singular Value Decomposition</td>
</tr>
<tr>
<td>Abbreviation</td>
<td>Description</td>
</tr>
<tr>
<td>--------------</td>
<td>----------------------------------------------</td>
</tr>
<tr>
<td>TSVQ</td>
<td>Tree Structured VQ</td>
</tr>
<tr>
<td>UTVQ</td>
<td>Unconstrained Tiling VQ</td>
</tr>
<tr>
<td>VBR</td>
<td>Variable Bit Rate</td>
</tr>
<tr>
<td>VLC</td>
<td>Variable Length Coding</td>
</tr>
<tr>
<td>VLD</td>
<td>Variable Length Decoding</td>
</tr>
<tr>
<td>VO</td>
<td>Video Object</td>
</tr>
<tr>
<td>VOL</td>
<td>Video Object Layer</td>
</tr>
<tr>
<td>VOP</td>
<td>Video Object Plane</td>
</tr>
<tr>
<td>VQ</td>
<td>Vector Quantisation</td>
</tr>
<tr>
<td>YUV</td>
<td>Luminance-chrominance colour space</td>
</tr>
</tbody>
</table>
Chapter 1

Introduction

1.1 Theme of the thesis

The rapid development of still and moving image compression standards (JPEG [jpeg], MPEG [mpeg], MPEG-II [mpeg2], H.261 [h261], H.263 [h263], H.263+ [h263p]) and their commercial success has created great interest in the issues of image and video compression techniques. While the problems of video compression for still images, CD-ROM video storage and television transmission seem to have been satisfactorily solved (at least for the time being), the area of interactive video communication (videophones, videoconferencing) still awaits a solution that would be acceptable for extremely low data-rates (below 10 kbit/s). While existing video coding standards can operate at low data-rates (below 64 kbit/s) their application to communication via PSTN lines and mobile links (where extremely low data rates are required) is limited by disturbing artefacts created by heavy quantisation. This situation calls for adaptation of an entirely different approach to analysis of a video sequence. Model based techniques offer such an approach.

This thesis seeks a solution to the problems of model fitting and model tracking in model-based coding of typical head-only and head-and-shoulders scenes. The fitting and tracking mechanisms are based on facial recognition techniques. Although videophone use is the primary application, the solutions proposed in this thesis could be adapted for use in other areas. These would include virtual speakers, videoconferencing, communication aids for the deaf, control aids for people with limb motion impairment and many others.

Since the system utilises a codebook of facial images for the purpose of automatic tracking and fitting of wire-frames in head-and-shoulders videophone scenes both the areas of video compression and facial recognition are relevant and have been thoroughly researched. The results of this research are followed by the introduction of a generic wire-frame fitting and wire-frame tracking algorithm.
1.2 Motivation

Despite the construction of many working models (AT&T, BT/Marconi, Comtech), a videophone capable of providing acceptable video quality at extremely low bit rates has still not been realised [li94]. The standard image resolution of approximately QCIF seems to be widely accepted but the frame rate is usually reduced to 5 per second. The application of motion compensated discrete cosine transform (MC-DCT) techniques in these devices adds block artefacts which are reported to be particularly annoying in the speaker's lips area. The recently developed H.263 standard (also based on MC-DCT) is optimised for bit rates of about 28.8 kbit/s (for operation with fast modems) [rijkse95]. When extended to extremely low bit rates, existing videophone systems are reported to produce very poor image quality [li94].

Compression of video using waveform based techniques (e.g. MPEG, MPEG-II, H.263) is the result of the removal of statistical, human visual system (HVS) and spatio-temporal redundancies. It seems that removal of these redundancies is not sufficient to achieve the compression ratios necessary for video transmission over extremely low bit rate links.

Semantic model based algorithms offer potentially better compression performance. This is because they target removal of semantic redundancy, i.e. they utilise information about the contents of the scene. The goal of semantic redundancy removal is achieved by application of machine vision techniques including motion tracking, object recognition and feature extraction. The actual scene is viewed here as a 2D projection of a 3D model, rather than a 2D waveform.

Although application of machine vision techniques might increase the computational load and complexity of the entire algorithm, the potential compression ratio offered by semantic model based techniques is much higher than that offered by waveform based algorithms. Since this thesis is concerned with analysis of typical videophone scenes, i.e. scenes with high semantic redundancy, the application of semantic model based techniques is very appropriate.
1.3 Thesis organisation

The remainder of this thesis is organised as follows.

Chapter 2 presents existing video coding techniques and their applicability for extremely low data rates. The techniques are subdivided into groups according to the mathematical model they utilise. Both waveform and model based techniques are described and compared.

Chapter 3 includes more detailed discussion of semantic based video techniques and the reasons behind their application to extremely low data-rates. Since the author believes that a strong connection between semantic analysis and facial recognition exists, the latter are also investigated. The technique most relevant to the proposed system is chosen. This follows a discussion of methods used for retrieval of the speaker’s silhouette, face and facial features.

In Chapter 4, a technique for automatic tracking of wire-frame vertices in head-and-shoulders scenes developed in this research is presented. Results of tests on widely used sequences are included. Comparison of the tracking of particular facial features is highlighted. A manually initialised automatic tracking system is described.

Chapter 5 describes a method of automatic wire-frame fitting (adaptation) developed during this research. This chapter includes a description of image segmentation techniques that extract facial features from the analysed scene. Techniques based on spatial correlation and principal component analysis are compared. The most promising technique is chosen for further analysis.

The order of in which the research was presented in Chapters 4 and 5 deserves explanation. Historically, some of the techniques presented in Chapter 5 were developed earlier than the PCA-based tracking algorithm (Chapter 4). However, it was the experience gathered during the development of the automatic tracking method, that allowed the creation of the most successful fitting algorithm (Chapter 5), i.e. the crucial sections of Chapter 5 are the results of research presented in the previous chapter. This also explains why some of the techniques presented in Chapter 5 are in fact less successful than those described in Chapter 4.

The techniques described in Chapter 4 and Chapter 5 are utilised for construction of a unified approach to automatic fitting and automatic tracking without human intervention. The above
and a comparison between manual and automatic initialisation of tracking in the sequence is the subject of Chapter 6.

Chapter 7 includes summary and conclusions along with insights into areas of future research. The latest developments in the area of extremely low data rate moving image coding are summarised. This chapter also includes discussion of real-time hardware applications.
2.1 Introduction

Compression of video for bit rates below 10 kbit/s is quite a daunting task. For high quality video data this requires a compression ratio of about 20,000 (Table 2.1). It is therefore necessary to have a very careful look at all existing still and moving image compression techniques and try to find one that might at least have the potential to achieve such a high compression ratio. In recent years the demand for compression of video for very low data-rates (below 64 kbit/s) has increased considerably thanks to the work of the MPEG-4 group [chiariglione97, sikora97]. Initially MPEG-4 was intended to be an encoding technique similar to MPEG and MPEG-II, but dedicated to very low data-rates. In the meantime the H.263 standard (targeting a data-rate of about 20 kbit/s) was proposed. This would have made MPEG-4 obsolete. The goal of MPEG-4 was redefined to make it a tool for multimedia (rather than audio and video only) coding. It would allow switching between the coding standards, depending on the available bandwidth. Apart from the existing techniques, the standard would be open for new waveform and model based algorithms. These would include algorithms for encoding video for extremely low data-rates.

<table>
<thead>
<tr>
<th>Video format</th>
<th>Data-Rate</th>
</tr>
</thead>
<tbody>
<tr>
<td>CCIR Rec. 601 (720 x 588 pixels)</td>
<td>~ 207 Mbit/s</td>
</tr>
<tr>
<td>CIF (360 x 288)</td>
<td>~ 37 Mbit/s</td>
</tr>
</tbody>
</table>

Table 2.1: Approximate data rates of an uncompressed video signal

The purpose of this chapter is to provide an insight into waveform and model based techniques and to expand on the issues raised above. The chapter also aims to justify the choice of a semantic approach as the most appropriate method for extremely low bit rate compression.
2.2 Existing moving image compression techniques

Current research in the area of very low data-rates (below 64 kbit/s) can be subdivided into two major groups: waveform based techniques and model-based techniques (Figure 2.1). While waveform based techniques treat an image as a two dimensional waveform, the model-based techniques view the image as a 2D projection of a real life 3D scene. Major standardisation efforts (MPEG, H.263) concern waveform based techniques only. The recently developed MPEG-IV standard (or rather its core: MSDL - MPEG Syntactic Description Language) allows the incorporation of new techniques (i.e. not standardised at the time of MSDL creation) including model-based techniques [mpeg4].

![Figure 2.1: Moving image coding techniques](image)

2.2.1 Waveform based approach

The main concern of waveform based video compression techniques is removal of spatio-temporal, human visual system (HVS) and statistical redundancies. The image is treated as a 2D matrix of sampled values (usually quantised at 256 levels). The following methods can be included in this group: transform (DCT) coding, fractal coding, wavelet (subband) coding and vector quantisation (VQ) coding. Since video coding techniques can also be presented as
a frame-by-frame still image coding (especially in the early stages of algorithm development where, for reasons of simplicity, motion compensation is not taken into account and only intra-frame data is considered) both still and moving image coding techniques will be considered.

2.2.1.1 Block DCT transform based coding

The compression techniques based on DCT are often referred to as ‘well-established’ since they have already found numerous commercial applications. Both still and moving image compression techniques based on DCT have been standardised. Since the techniques share much common ground, only MPEG will be briefly described here.

MPEG is a block-based technique. Each frame is logically subdivided into ‘blocks’ of 8×8 pixels in the YUV domain. Four blocks of luminance (Y) form a 16×16 pixels ‘macroblock’. While the DCT is performed on blocks, motion compensation is performed on macroblocks. MPEG defines four types of frames: intra-coded (I-frames), predictive-coded (P-frames), bidirectionally-predictive-coded (B-frames) and fast-forward playback mode (D-frames). The D-frames take no part in motion compensation. The I-frames are coded without reference to the content of other frames. While the P-frames are encoded using motion compensation techniques with reference to the past I- or P-frames only, the B-frames are encoded using motion compensation with reference to both past and (or) future I- and (or) P-frame(s). This results in the highest and lowest compression ratios for the B- and I-frames respectively. Motion compensation is carried out by matching of the luminance macroblocks of the current (analysed) frame to the luminance macroblocks in the reference frame within a certain search region. The macroblock from the reference frame which is most ‘similar’ to the analysed macroblock from the current frame defines the direction of the motion vector. The difference between the current macroblock and most ‘similar’ macroblock from the previous frame (also referred to as the ‘predicted’ macroblock) is encoded along with its motion vector. The measure of similarity between the current and predicted macroblocks is a function of a difference between the two frames (2.1).

\[
D(i, j) = \frac{1}{MN} \sum_{m=1}^{M} \sum_{n=1}^{N} f((l(m, n) - l(m + i, n + j)))
\]  

(2.1)
where \( n \) and \( m \) denote pixel indexes, \( l(m, n) \) - original macroblock pixel at \((m, n)\), \( l_r(m, n) \) - reference macroblock pixel at \((m, n)\), \( M \) and \( N \) are the macroblock dimensions, \( i \) and \( j \) are the co-ordinates of the motion vector (displacement). The \( f(x) \) function (also referred to as distortion measure) can be defined either as \( MAD \) (Mean Absolute Difference) (2.2) or \( MSD \) (Mean Square Difference) (2.3).

\[
MAD (i,j) = \frac{1}{MN} \sum_{m=1}^{M} \sum_{n=1}^{N} | l(m,n) - l_r(m+i,n+j) | \tag{2.2}
\]

\[
MSD (i,j) = \frac{1}{MN} \sum_{m=1}^{M} \sum_{n=1}^{N} ( l(m,n) - l_r(m+i,n+j) )^2 \tag{2.3}
\]

A set of macroblocks following each other in the bitstream creates a slice. The slice can begin at any macroblock boundary. The first slice must begin with the first macroblock of the picture (top-left corner). The last slice must end with the last macroblock of the picture (bottom-right corner). Each macroblock may belong to only one slice. Slices form a frame. Frames (I, P and B) are organised in groups of pictures (GOP). If B-frames are used the GOP order of the frames is different in bitstream (bitstream order) and after decoding (display order). This is because in order to decode a B-frame both past and future frames are used. A set of GOPs creates a sequence. The sequence may consist of one or more GOPs. Suppose the order of frames in a GOP as follows: IBBPBBP. Figure 2.2 shows which frames in the GOP are used for prediction of P- and B-frames.

The P1 picture is predicted using motion compensation from the I picture, similarly the P2 picture is constructed basing on the P1 picture. These three pictures (I, P1, P2) are referred to as ‘anchor pictures’ [chiang94]. B pictures are derived from the closest past and future anchor pictures (e.g. B2 is derived from I and P1, B4 is derived from P1 and P2). All pictures, apart from the I type (intraframe coding), are coded as a difference between prediction (motion compensated) and original (interframe coding). Once the difference between original and predicted picture is calculated (in the case of B and P pictures), the resulting signal is applied to the DCT encoder. Here the basic unit of a picture is a block (8x8 pixels), not a macroblock. The DCT encoder produces another block containing DCT.
coefficients. The content of the resulting macroblock is scanned in 'zig-zag' order (Figure 2.3) and quantised.

Each DCT coefficient may have different quantisation levels. The coefficient-specific quantisation level is stored in an $8 \times 8$ quantisation matrix. There are different quantisation matrices for I-picture macroblocks (intraframe quantisation matrix) and B or P-picture (differential) macroblocks (non-intraframe quantisation matrix). Although quantisation matrices of both types are suggested in MPEG standard, they may be altered by the software developer. In general, quantisation levels should increase with the frequency that is represented by a specific DCT coefficient. This is because the human visual system is more sensitive to lower frequencies.
The quantised DCT coefficients are finally coded using variable length codes (VLC). The variable length table used to code blocks of DCT coefficients favours the situation when most of the ac coefficients (all coefficients in a block apart from the top-left corner one, which is a dc-coefficient) are zero by assigning shorter codewords. Before the encoding process starts I-, P- and B- frames are re-ordered. Thus IBBPBBP (GOP order) is changed into IPBBPBB (encoding order) to allow the decoding of P-frames in advance of B-frames. The task of the motion estimator (Figure 2.4) is to find a motion vector for the specific macroblock. Motion vectors are then passed to the VLC (variable length) coder and multiplexed with coded macroblocks to create a MPEG-compliant (VBR) bitstream. The bitstream is then passed through a buffer, which is supposed to produce a constant bit rate stream (CBR) - still compliant with MPEG. The fullness of the buffer must be constantly controlled and the quantisation scale must be adjusted accordingly. The issues of buffer occupancy are addressed in [saw97].

While the structure of MPEG allows certain parts of the encoder to vary (e.g. application of different block matching techniques would result in calculation of different motion vectors), the shape of the MPEG decoder is almost determined by the structure of the video bitstream (Figure 2.5).

If the input stream is CBR, then minimum size of input buffer is easy to calculate. If, however the bit rate is variable the size of buffer cannot be calculated using data stored in the incoming bitstream. An input buffer is still necessary here, as the bit rate required by the decoder (decoding speed) is not the same as the input bit rate. After passing through the input buffer the bitstream is de-multiplexed in order to separate coded blocks and macroblock
motion vectors. Data is then VLC-decoded and the block stream is additionally passed to a
de-quantiser and module performing inverse cosine transform. Decoded motion vectors are
used to recover motion compensated pictures in the following way. After a reference picture
(e.g. I picture) is decoded it is stored in the frame memory. Subsequently a motion
compensated picture (e.g. P picture) is restored by the predictor module. Finally the order of
the pictures is restored.

There is a number of issues that the MPEG video encoding technique does not specify. It is
up to the designer of the decoder which type of picture to use for each incoming frame or
what motion vectors to use for each macroblock. The motion estimation of an individual
macroblock in a B picture can be carried out based on past, future, or both pictures. The
quantisation scale can be changed for any slice (it is not possible to change quantisation scale
for each slice unless the slice consists of one macroblock only). Each macroblock can be
coded using intra-coding only, even if it is part of P or B picture. A macroblock coding may
be skipped if the predicted macroblock does not substantially differ from the original
macroblock. Although suggestions as to the shape of the intra- and interframe matrices are
given, they can be altered for each sequence (it is not possible to change matrices for each
picture unless the sequence consists of only one GOP containing one I frame). Sequence,
GOP, frame, slice, macroblock and block are "layers" of moving image. Table 2.2
summarises what features of moving image are set on each level.

<table>
<thead>
<tr>
<th>Layer</th>
<th>Feature</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sequence</td>
<td>Horizontal / vertical size, pixel aspect ratio, picture rate, decoder</td>
</tr>
<tr>
<td></td>
<td>buffer size (CBR only), constrained parameters, intra quantiser</td>
</tr>
<tr>
<td></td>
<td>matrix, non-intra quantiser matrix</td>
</tr>
<tr>
<td>GOP</td>
<td>time code, closed GOP, broken link (Closed GOP only)</td>
</tr>
<tr>
<td>Picture</td>
<td>order in GOP, decoder buffer flush time, forward motion vector</td>
</tr>
<tr>
<td></td>
<td>precision and code (not in I pictures), backward motion vector</td>
</tr>
<tr>
<td></td>
<td>precision and code (not in I nor P pictures)</td>
</tr>
<tr>
<td>Slice</td>
<td>Start address, quantiser scale</td>
</tr>
<tr>
<td>Macroblock</td>
<td>Start address, type, motion vector</td>
</tr>
<tr>
<td>Block</td>
<td>Coded DCT coefficients</td>
</tr>
</tbody>
</table>

Table 2.2: MPEG control levels
If a feature is set on certain level (e.g. sequence) then all the levels below it (i.e. GOP, picture, slice, macroblock, block) inherit that feature. The feature is valid until cancelled by another part of bitstream of the same level.

The issue of the application of block-based DCT techniques for very low bit rate video coding is re-visited very often. Most recently on the occasion of their application to mobile communication: [cherriman96], [gharavi96], [faeber96]. Although certain successes were reported, there seems to be wide agreement, that at the present moment the systems based on block DCT cannot go below certain limits as far as the data-rate is concerned. The level of control (Table 2.2) available to the constructor of the system is not sufficient to ensure good image quality at extremely low data-rates. There seem to be numerous ways of increasing the quality of the image compression, probably the most radical [li94] proposes abandoning the structure of the closed-loop encoder (Figure 2.4). If certain pattern recognition techniques were used for detection of the human face (e.g. in the case of ‘head-and-shoulders’ scenes), it should be possible to vary the quantisation step according to the image contents. This should allow at least improving the quality of video encoding in the area occupied by the lips. Although the problem has been studied, it was not included in the creation of new standards ([eleftheriadis95], [eleftheriadis95-2]). Thus, it can be said, that so far, attempts to create an extremely low bit rate moving image codec based on DCT techniques have failed. Other techniques and their contribution towards decreasing the bit rate must be examined.

2.2.1.2 Vector quantisation based coding

Perhaps the most straightforward technique of image encoding is that based on vector (or block) quantisation. The intention of the vector quantisation (VQ) technique (originally presented in [gersho82] and [gray84], then developed in [gersho93]) is to represent a signal by an optimised codebook. A simple example gives a good insight into the problems of the vector quantisation approach. Let us consider a 32 level greyscale image of size 128x128 pixels. The total number of bits describing this image is 128x128x5 (since 5 bits are sufficient to describe 32 levels). This gives a total of $N = 81920$ bits per image. Suppose the image is now subdivided into non-overlapping blocks of 4x4 pixels. This would result in 1024 such blocks. Since there are 16, 5 bit pixels in each block there are $32^{16}$ combinations of its texture. This would require a coding word, or index $\log_2(32^{16}) = 16 \times 5 = 80$ bits long. If instead of sending the image pixel by pixel we decided to send the index for each block we
would have to send \( L = 80 \times 1024 = 81920 \) such indices. Assuming that the codebook is known by the encoder and the decoder, the decoding process would be almost instantaneous, since it would involve re-calling the appropriate entry from the cookbook. However at this stage, since \( N = L \), no compression takes place. However if it was possible to reduce the number of blocks in the codebook (i.e. to reduce the length of the index) compression would be possible. The vector quantisation method is very attractive, since the decoding process is extremely fast. It involves accessing the codebook via a lookup table so construction of dedicated hardware becomes unnecessary. However, in order to compress the image effectively, an appropriate codebook must be chosen, to minimise both its size and the distortion in the encoded image. As a matter of fact, the choice of codebook is the most difficult problem in vector quantisation. Reduction of the codebook size also means that for each block a search of the most appropriate (the most similar) representative must be performed. This adds to the computational costs of encoding.

There are many types of vector quantisation techniques. Each of them copes differently with the problems of codebook generation or codebook search. In order to reduce the computational costs of search a tree-structured approach (TSVQ) is often used. In its simplest form it uses binary tree search [kossentini92, chou89, gray82]. The encoder steps down node-by-node until the lower end of the tree is reached (Figure 2.6). Since at each node the block most similar to the unknown is chosen, it is likely that the end-node of the branch is the best approximation.

![Figure 2.6: Binary split in TSVQ](image)

In order to maximise the possibility that the end node is actually the best approximation of the encoded block (or at least is relatively close to its best approximation), rules based on probability density function are applied. They favour access to elements of the codebook that are more likely to appear among the unknown (analysed) blocks. These systems are also referred to as lattice VQ systems. In MSVQ (Multi-Step Vector Quantisation) an unknown block is encoded using a relatively small codebook. Once the best approximation of the unknown block is found, the difference between the two is encoded using another codebook.
This process can subsequently be repeated yielding an improved approximation of the true value at each stage [hammer87, ho88].

So called ‘product codes’ are a variation of the vector quantisation technique in which the block is quantised in two parts (i.e. two codes are assigned to it) and the reconstruction is achieved by combination (Cartesian product) of the indices assigned to each part. An example of product code is MRVQ (Mean/Residual VQ). In MRVQ the two parts are the mean of the block (i.e. scalar) and the mean adjusted unknown block.

Probably the most widely known VQ technique is hierarchical VQ. Unlike in the previously presented methods, the image is first analysed in order to separate areas of higher spatial frequency (high details) from areas of lower spatial frequency (homogenous, low spatial detail). The high/low frequency areas are identified by a quad-split performed on the entire image. A larger sub-image is subdivided into four smaller ones of equal size if certain homogeneity criteria are not satisfied. The criteria are usually based on the mean value or the dynamic range of greyscales in the analysed sub-image. Once found homogenous (i.e. once no further split is possible) the sub-images may be encoded using earlier mentioned techniques [daly88, nasrabadi88]. Hierarchical VQ techniques have many variations. These include MSHVQ (multi-stage hierarchical VQ) [ho88] in which a quad-tree based on image differences is created (i.e. once the decision to split is made the mean of the parent sub-image is subtracted from the resulting sub-images), UTVQ (unconstrained tiling VQ [boxermann90]) in which regions are not constrained to be a part of a quad-tree or even to be rectangular-shaped [cortereal90].

Adaptive techniques constitute a separate group of vector quantisers. A simple example of adaptive VQ would be FSVQ (finite state VQ). Here the choice of applied codebook is dependent upon the analysed block (vector). The codebook may be chosen based on texture content of the analysed vector or based on the output from the analysis of a similar or even previously chosen block. A ‘dynamic’ adaptive VQ technique (DFSVQ) presented in [nasrabadi90] assumes existence of only one codebook. The codebook would be dynamically rearranged so that codewords (codebook elements) most similar to the analysed one can be found faster.

Although the research on vector quantisation is very much alive, there seems to be no immediate application in extremely low bit rate image and video coding. As a matter of fact
there is no contribution suggesting application of VQ based technique at data rates lower than 10 kbit/s. Efforts in the VQ area seems to be concerning problems of codebook design and search methods, rather than reduction of the bit rate. Obviously the latter would be an inherent feature if any improvement (as far as compression ratio is concerned) was achieved, but at this stage the compression results seem to be poorer than those of wavelet-based systems.

2.2.1.3 Fractal based coding

In recent years fractals have gained considerable attention because of their ability to describe real life objects using mathematical models [falconer90]. The term ‘fractal’ is used very often and on various occasions. A fractal is a function. Mathematics discarded fractals for a very long time as a set of functions not worth looking at. Fractals differ from other types of functions, but it is not possible to set a boundary and simply classify functions as ‘conventional’ and ‘fractal’. In general fractals can be described as ‘irregular’ and ‘non-smooth’ functions. Fractals are simultaneously extremely complicated visually and very simple mathematically. Fractals are built either of themselves or some parts of themselves. The other, better-known feature of fractals is their graphic representation (often referred to as a ‘fractal image’). It has been noticed that it is possible to construct almost real-life looking objects using a very limited set of equations. One of the best known fractal images is a fern. It is also possible to generate much more complicated images. ‘Expansion’ ratios in range of 10000:1 are possible using fractals. The word ‘expansion’ is used on purpose here, because the compression of a real-life image is a totally different matter.

The fundamental work on fractal compression was presented relatively recently [barnsley88]. Unfortunately it was patented [barnsley90] and its availability to the scientific community is limited. An independent fractal encoding technique was also proposed in [jacquin92]. In contrast to [barnsley88] the description of the proposed technique is available to the community of researchers working in the area of image and video compression. The two methods differ from each other in the sense that the one proposed in [barnsley88] is not ‘automatic’. This means that the method relies on the use of certain operators to find correspondences between various parts of the image [bedford94]. The technique proposed in [jacquin92] proposes ‘automatic’ encoding of an image. Although different, both approaches are based on common mathematical theory (Figure 2.7).
Fractal based compression promises enormous advantages over existing techniques. The algorithm (at least in theory) promises the following features:

- compression ratio of up to 10000:1,
- real time encoding and decoding using software tools only,
- better picture quality than the well-established techniques (e.g. block based DCT)

Since the expected compression ratio would be appropriate for extremely low bit-rate applications, the fractal based approach deserves a closer look.

An example of application of fractals for video coding can be found in [lazar94] (most of the research published after 1992 is based in one way or another on [jacquin92] or [barnsley90]). Fractal compression was proposed for both rectangles [jacquin92] and regions of irregular shape [thomas95].

Fractal encoding algorithms are all built on an assumption, that the image analysed (e.g. grey-scale) is a fractal. Let us consider the following function (Figure 2.8). The y-axis corresponds to a grey-level and the x-axis describes the position of a pixel in a certain line of an image. The fractal encoding of the image (in the form proposed in [jacquin92]) can be described as follows. Let us consider two windows for a given line of the image: 'range block' (smaller
window) and 'domain block' (bigger window). The purpose of the algorithm is to find a mapping function that would transform the domain window onto the range window with the smallest error. The number of transformations that can assure acceptable transformation is limited (this reduces computational effort but, as reported, still produces very good results). As it can be seen in Figure 2.8, flipping the domain block horizontally and then scaling it down by some factor might produce a certain approximation of the range block.

![Figure 2.8: Fractal analysis of an image](image)

However, this approximation might not be the best one (one of the problems of fractal coding concerns finding out which transformation is optimal - with regard to quality or compression ratio). Once the best approximations for all the range blocks are found, the new image is called a 'collage' of the original. The overall transform $T$ (i.e. the transform which is a piecewise sum of the local transforms that were used for each range block) is the key to rebuilding the image. It is therefore sufficient to encode the transform $T$ instead of the entire image.

Although the theory shows that the number of iterations required to rebuild the original image may be infinite, the number of iterations necessary to rebuilt a $256 \times 256$ grey-scale image with acceptable results, is about 9.

As proposed in [jacquin92] the size of a domain block should be four times area of the corresponding range block and it may be transformed using one of the 8 spatial transformations (isometries) and one of the 4 grey-scale transformations. The allowable spatial transformations are:
• Identity (the domain block is mapped onto the range block without changes).
• Orthogonal reflection about vertical axis (the domain block is mapped onto the range block after it is flipped vertically).
• Orthogonal reflection about horizontal axis (the domain block is mapped onto the range block after it is flipped horizontally).
• Orthogonal reflection about first diagonal (the domain block is mapped onto the range block after it is flipped about axis connecting left top and bottom right corner).
• Orthogonal reflection about second diagonal (the domain block is mapped onto the range block after it is flipped about axis connecting bottom left and right top corner).
• Rotation: +90°, +180°, -90°.

The allowable grey-scale transformations are: luminance shift, contrast scaling, colour reversal, and absorption of grey-level. Encoding of the image can be divided into two steps. Each step is repeated for each range block within the image (the range blocks do not overlap each other). In the first step the domain block is selected for a given range block. In the second step, the best transformation (the one producing the smallest error) is selected and applied. The domain block for a given range block can be placed anywhere in the image. This might make the selection of the domain block quite a long procedure. In practice, the domain block is searched for in steps (usually the step is equal to the side of the range block or half of the side of range block). The set of domain regions that will be processed further is called the ‘pool of domain blocks’. After the domain pool is chosen it is classified into three non-overlapping sets of blocks basing on their perceptual features (Figure 2.9).

Figure 2.9: Domain blocks classification

The smooth (also called ‘shade’) blocks contain no significant edges or textures. The edge blocks contain a sharp change of intensity across a curve. The midrange (also referred to as textured) block is assumed to be isotropic. The edge blocks are further subdivided into simple edge blocks and mixed edge blocks (Figure 2.9). Only the edge blocks and midrange blocks are used in further analysis. In the second stage the range blocks are analysed. They are classified using the same criteria as domain blocks. If the range block is a smooth block, it is approximated with the uniform grey level equal to the average value of the analysed block. If
the block is a midrange block, the transformations are also restricted to those from the grey-scale group. Here luminance shift and contrast scaling is suggested as the most appropriate transformation. In case of edge blocks, both spatial and grey-level transformations are taken into account. Finally one of the 8 isometries described earlier is used. After the domain block is chosen and transformed, it is called the ‘matching block’. It is worthy noticing at this point that the size of the range blocks does not have to be uniform. Allowing the size of the range block to vary makes the encoding technique more flexible. The image may be split into the range blocks using the quad-tree. In fact the same or similar quad-tree split may be used in VQ based techniques. As we learn from the following table certain features are shared by fractal and VQ based codecs (Table 2.3).

<table>
<thead>
<tr>
<th>Vector quantisation</th>
<th>Fractal block coding</th>
</tr>
</thead>
<tbody>
<tr>
<td>Code-book based on set of images, the code-book is sent off-line</td>
<td>Pool of domain blocks, processing of the pool, no transmission of the pool necessary</td>
</tr>
<tr>
<td>Block matching of the current block to the block in the code-book</td>
<td>Block matching of the range block to the best processed domain block</td>
</tr>
<tr>
<td>Decoding of the image is based on look-up table (look-up table is transmitted as a online bit stream)</td>
<td>Decoding of the image is based on iteration using the formulas derived during compression</td>
</tr>
<tr>
<td>Different block sizes allowed, quad-tree split used</td>
<td>Different sizes of the range blocks allowed, quad tree split used</td>
</tr>
</tbody>
</table>

Table 2.3: Fractal block coding versus vector quantisation

In fact, the two techniques are very similar as far as general idea of encoding is concerned: the images are encoded using another set of images (reference images). The reference images are not sent on-line. In the case of vector quantisation, they are sent off-line and in the case of fractal block coding, the code-book is not sent at all, because the image itself can be regarded as the code-book. The major difference is obviously the mathematical model.

Fractal block coding was reported to outperform the ‘state-of-the-art’ vector quantisation coders [jacquin93]. There are however still certain problems, mainly associated with the following issues:
Chapter 2: Moving image compression systems

- 'Blocky' artefacts are still visible, it could be worthy investigating a different shape of the range block (the 'block' could be in fact replaced by the 'region').
- The range of area to seek for best match for the given range block is unclear.
- Overlapping of the range blocks could be allowed.
- The achieved compression coefficients are far from 10000:1.

In 'standard' form the technique presented in [jacquin92] is based strictly on pre-segmentation of the image. That may seem as a good approach for reduction of computational complexity, but it also seems that the algorithm presented suffers from to close similarity to the existing and well-established techniques (block-matching, segmentation using quad-tree split). This technique should in fact be called 'fractal block coding' rather than 'fractal' coding, because of the significant influence of block-based techniques.

The method from [hurd92] proposes application of a fractal based codec for coding of video. Decompression of realistic video was claimed to be possible using an IBM AT 80286 (!) based computer. The description of this method ([hurd92]) is very short, and lacks detail. Similarly to the still image coding this invention is protected by a patent [barnsley] and is not accessible to the scientific community. The only useful information presented is the compression performance of the proposed system (Table 2.4).

<table>
<thead>
<tr>
<th>Compression ratio</th>
<th>bytes/frame</th>
<th>PSNR</th>
</tr>
</thead>
<tbody>
<tr>
<td>79:1</td>
<td>243</td>
<td>30.8</td>
</tr>
<tr>
<td>58:1</td>
<td>332</td>
<td>32.6</td>
</tr>
<tr>
<td>21:1</td>
<td>930</td>
<td>39.2</td>
</tr>
</tbody>
</table>

Table 2.4: Results of fractal compression applied to video sequences

Because the method using fractal compression proposed in [hurd92] is not available it is necessary to refer to other sources for explanations. In fact, a video compression algorithm using fractals has also been proposed in [lazar94]. It differs from the algorithm proposed in [hurd92] significantly in the sense that the fractal video coder proposed is an inter-frame coder. The proposed algorithm uses 3D range and domain blocks (the terms 'range' and 'domain' have been previously defined for a static image - Figure 2.8). Range blocks are created from range frames (R-frames). R-frames are consecutive, non-overlapping groups of frames (Figure 2.10).
Similarly to the 2-dimensional case, there are domain-frames (D-frames) associated to the R-frames (Figure 2.11). The operations during 3D compression using a fractal-block method are very similar to those performed in the 2D case. However, in 3D space the number of possible isometries is considerably larger due to the fact that a 3D spatio-temporal block is not isotropic.

The changes of luminance within the same frame are far more radical than the changes that can be observed at the same spatial position, but with changing time. This is because the motion within the frame sequence is usually slow. It was therefore decided that each domain
block may only be transformed by two types of isometries: intra-frame isometry and inter-frame isometry. The set of proposed intra-frame isometries is identical to those proposed in [jacquin92]. The set of inter-frame isometries was considerably reduced. Only two inter-frame isometries are allowable: identity and time reverse. Similarly to the two-dimensional case, the most computationally expensive procedure is the search for the best 3D domain block for a given 3D range block. Full search seems to be impossible to implement without the application of an array of processors. It was therefore decided that only the 'nearby' domain candidates would be tested. The choice of segmentation of the R-frame into the range blocks was also inspired by the two-dimensional case. A three-dimensional case of the quad-split algorithm is called 'octal-split'. Also a new spatio-temporal split method was proposed in [hurd92]. A 3D block can be split into four or two range regions only. The split into two range regions is only allowed in the temporal dimension. The split into four range regions is only allowed in the spatial dimension (Figure 2.12).

![Figure 2.12: A novel spatio-temporal split](image)

Tests carried out on the Miss America sequence showed that encoding quality is heavily dependent on the size of the domain regions pool [hurd92]. Also the novel split technique was compared with the octal-split. The results show that the traditional split technique yields better compression ratio, but the quality of the encoding improved when the novel split technique was applied. The results reported were quite encouraging. The average compression ratio was between 40 and 77 (which is comparable with the compression ratio achieved by MPEG).

An original algorithm was proposed in [zhao92]. The fractal functions were combined there with a DCT. The image was first segmented into two kinds of blocks: $8 \times 8$ and $16 \times 16$ pixels. Those blocks are subsequently transformed using DCT and whole fractal analysis.
takes place in the frequency domain. Both range blocks and domain blocks have fixed dimensions (8x8 and 16x16 coefficients respectively). The range blocks are further classified into two groups: simple range blocks and complicated range blocks. The basis for classification is the absolute value of the sum of the three ac coefficients \( l(i,j) \) surrounding the dc coefficient (2.4).

\[
T = |l(0, 1) + l(1, 1) + l(1, 0)|
\]  

(2.4)

If the range block is ‘simple’, then it is approximated by storing its dc coefficient only. If the range block is ‘complicated’ it is encoded using similar tools to those that were described earlier on. The reported compression ratios are higher than those reported using fractal functions in the spatial domain, but it was admitted that more details were lost using that method.

In [thomas95] the image is segmented into regions rather than squares, before the fractal theory is applied. Here the basic ‘unit’ is referred to as a ‘range region’ consisting of range blocks (squares of 8x8 pixels). For each range region there is a domain region. In contrast to the range regions, the domain regions can overlap each other (the range regions cannot overlap each other and their graphical sum must result in reconstruction of the analysed image). That allows the choice of the best domain region that would be subsequently mapped onto the corresponding range region. In this method two grey-scale transformations are allowed: contrast scaling and luminance shift. The spatial transformations (isometries) described in [jacquin92] were also allowed here without any modifications. The parameters describing the isometries and grey-scale transformations were subsequently encoded using a Huffman table in order to increase the compression ratio. Segmentation of the image is carried out using a derivative of the region growing technique. First the entire image is split into 8 x 8 squares (Figure 2.13). An arbitrary square is chosen to be a ‘seed’. It is subsequently treated as range block and the algorithm presented for block based fractal transform is applied for this range block. Then the region block is extended in one of four directions (joint range block). For each direction the same parameters that were found to be optimal for the region block are analysed for the joint range block, now using the joint domain block (Figure 2.13). If processing using the same isometries on the joint range and domain blocks is more efficient, the two original range block are connected. The same procedure is repeated until there is no other range block that could be connected to the analysed range block. A new seed is found after the analysed region cannot be extended and
the same algorithm is repeated until the entire area of the image is analysed. Using this technique, the number of transformations which must be performed on a single image can be reduced by up to 20 times. This can increase the compression ratio considerably along with the speed of decoding. The reported compression ratios (at reasonable signal-to-noise ratios) are very promising and range from 150:1 to 180:1.

It is fair to say, that the development in the area of fractal based image and video encoding has been remarkable in recent years. However fractal based techniques, so far, fail to deliver on promises of 10000:1 compression ratios. As a matter of fact the compression ratio is comparable to previously described techniques.

2.2.1.4 Wavelet based coding

Wavelet techniques were proposed for moving image compression only recently. The wavelet transform was introduced to image processing in [mallat89]. The theory was further clarified in [daubechies90]. The first application of wavelets to image compression can be found in [antonini92]. The wavelet transform enables representation of a finite energy function \( f(x) \) (2.5) in terms of a set of functions (also referred to as the ‘basis functions’) which are scaled (dilated) and translated versions of the single function (also referred to as the ‘mother function’): \( \phi(x) \) (2.6). In most cases, the basis functions are scaled by powers of 2. In this case, the functions \( f(x) \) can be represented in the following form.

\[
f(x) = \sum_{m} \sum_{n} c_{mn} \phi_{mn}(x) \quad (2.5)
\]

\[
\phi_{mn}(x) = 2^{-m/2} \phi(2^{-m} x - n) \quad (2.6)
\]
The wavelet transform is the set of $c_{mn}$ coefficients with two parameters: position ($n$) and scale ($m$). It is argued, that for some functions, wavelets offer better approximation than the Fourier transform. The main difference between the wavelet transform and the Fourier transform is that the former tends to describe the signal in terms of the scale rather than the frequency. The usefulness of the wavelet transform for image processing comes from the fact that the images can be thought of as two-dimensional signals containing similar features at different scales (this feature of images is also used in fractal compression techniques). A good example could be the way in which the wavelets describe an edge on the image. Suppose a square window containing an edge was chosen from the analysed image. The energy of the functions used in Fourier decomposition would be distributed evenly over the entire window. The wavelets (i.e. the functions used in wavelet decomposition) behave differently. The energy of the wavelets is concentrated in the area of the edge. What it means in practice, is that the wavelet transform is scale invariant (a feature which is not clear in the case of the Fourier transform). Wavelets that are used to describe a particular function are derived from another function. The number of researchers working on the wavelet transform is increasing. This is because, as reported, the error of reconstruction of an image that had been compressed using the wavelet transform is smaller than the error of reconstruction of the image that had been encoded using DCT-based techniques (i.e. MPEG with the same compression ratio). Wavelet-based encoders are also relatively easy to implement. What is probably the most important from the point of view of extremely low data rates is that the blocking artefacts, although still present, are less annoying for the viewer.

Wavelet based video coding techniques were the first serious challenge to the 'establishment' of the DCT transform based standards (MPEG, MPEG-II, H.263). The MPEG-4 forum provided an open forum at which the wavelet-based moving image techniques were shown to outperform DCT based ones [martucci97, zhang92]. The structure of a wavelet based motion-compensated encoder is similar to that of DCT based ones (Figure 2.14).

Although the wavelet transform has a global nature, the motion compensation is still carried out based on blocks. In order to reduce artefacts that might arise from the block nature of motion compensation, overlapping was allowed. In [martucci97] the block-based motion estimation was actually adopted from H.263 without any major changes. While high compression ratios in DCT transforms result in blockiness, they show themselves in the form of 'ringing' artefacts if high quantisation is applied in the wavelet domain. The ringing effect
may be reduced by the application of different filter lengths at different levels of decomposition (e.g.: at the start of decomposition, longer filters are used to avoid any artificial blockiness while the use of shorter filters at the end of decomposition assures reduction of ringing). Other artefacts due to coarseness of quantisation of wavelet coefficients include blotchiness (fuzziness) in flat areas and reduction in thickness or even elimination of edges (which might be identified with loss of high frequency information).

![Diagram of motion compensated wavelet based moving image encoder](image)

**DWT:** Discrete Wavelet Transform  
**1/DWT:** Inverse DWT  
**AC:** Arithmetic Coding  
**FB:** Frame Buffer  
**H:** Huffman Coding  
**BME:** Block Motion Estimation  
**IQ:** Inverse Quantisation  
**OBMC:** Overlapping Block Motion Compensation  
**BF:** Output Buffer  
**R:** Picture Reorder

*Figure 2.14: Motion compensated wavelet based moving image encoder*

A simple example of a wavelet transform applied to an image is presented in [lewis92]. Four subbands (one low-pass and three high-pass) are created (Figure 2.15).

![Principle of wavelet decomposition](image)

*Figure 2.15: Principle of wavelet decomposition*
Although the four images created as a result of the first decomposition are subsampled (i.e. the number of pixels has been reduced by a factor of 4), the data is preserved, because each subimage contains a filtered version of the original. Filtering may be subsequently applied to the low-pass output from the first filtering stage. This results in another four subimages. This process can be repeated again until the desired level of decomposition is achieved (Figure 2.15).

The nature of wavelet coding makes it almost a perfect tool for progressive transmission of images (and video) i.e. when an image is sent in coarse-to-accurate mode (e.g. in web browsers). Still, its application for extremely low bit rate encoding is questionable. The compression ratios, although better than those obtained using DCT-based techniques, are still comparable. Wavelets do not seem to be a perfect tool for application to extremely low bit rate moving image encoding techniques. In fact, numerous publications suggest that wavelet methods are more likely to be a very good tool for encoding high quality images and a possible future replacement of block DCT based methods [sampson94, dasilva94, albanesi94] thanks to their ability to reduce blocky artefacts [ohta93, yao93] - a common problem with MPEG or H.263.

2.2.2 Model based approach

Model-based image coding techniques are often referred to as “second-generation”. These techniques view an image (a frame of a certain sequence) as a 2D projection of a 3D scene. Model based techniques are usually subdivided into two groups: object-oriented techniques and semantic-based techniques. Although both utilise models for description of the scene, they are quite distinct and a different set of problems can be associated with each of them.

2.2.2.1 Object oriented coding

Object-oriented interpretation of the scene was proposed first in [musmann89] in the form of a system called OBASC (Object-Oriented Analysis-Synthesis Coding). The technique attempts to segment the scene into non-rectangular segments representing objects performing separate motion in the scene. Since the shape of the object is not known a priori, additional data must be sent in order to describe its shape. In the OBASC system, each object is described by three parameters: its motion, shape and colour. Encoding of the moving video in
OBASC involves deriving the three parameters for each object. Three sets of parameters form the encoded OBASC bitstream (the motion set: A, the shape set: M and the colour set: S). The colour information consists of luminance and chrominance parts. Although the encoder in model-based coding looks familiar (a closed loop system) there are some quite significant differences. Once of the most important is inclusion of a 'source model' (Figure 2.16)

![Diagram](image)

**Figure 2.16: Encoder in the OBASC system**

The choice of the source model influences the analysis of the motion in the scene. In [musmann89] two source models were considered. A model assuming that moving objects are 2D (planar) rigid elements moving in the 3D world, and a source model assuming that the moving objects are 3D rigid elements moving in the 3D world.

The encoding process in the OBASC system (Figure 2.16) can be summarised as follows. The incoming frame is analysed in the 'image analysis' block. During analysis the frame is subdivided into objects. Each object is then described by a set of parameters A (motion), M (shape) and S (colour). The analysis is carried out using the reconstructed previous frame and the source model. The previous frame is reconstructed in the encoder by the local parameter decoder. The local decoder uses parameters stored in local memory. These parameters are stored in the memory as a result of analysis of the previous frame. The same parameters will have been sent in the form of an encoded bitstream via the transmission channel. The local memory is different from the frame memory in block-based encoding techniques, in that it stores A, M and S parameters, not the frame itself.
One of the most important (and new) issues in OBASC is the shape encoding algorithm. Fourier descriptors, polygon approximation and polygon-spline approximation were considered as candidates. Fourier descriptors fail in areas of sharp shape changes (corners). On the other hand polygon-approximated shapes look artificial. The polygon-spline approximation was proposed in [hotter90] as an improvement on polygon approximation.

Before the shape encoding algorithm can be applied, the shape itself must be extracted from the frame, or in other words, the frame must first be subdivided into individual objects. In [musmann89] it is assumed that the frame is subdivided into objects based on the amount of movement of a certain area of the frame. The segmentation algorithm (called a ‘hierarchical image analysis algorithm’) attempts segmentation of the scene based on the amount of motion performed by individual ‘objects’ (e.g. in the case of a head-and-shoulders scenes - the speaker’s silhouette, the speaker’s face and facial features). At the beginning the entire frame is viewed as a single stationary 2D object (when 2D planar objects in a 3D world are considered). The above assumption is subsequently verified by comparing the current frame to the previous frame (or frames) in the sequence. The areas where the frame difference is non-zero are potential locations of moving objects. Based on the frame difference and the assumed model, the image of the current frame is synthesised from the previous frame and compared to the ‘real’ current frame. The assumption that the model is able to describe the motion between the two frames - in the sense that there are no other moving objects in front of the analysed one - is thus verified. Otherwise there must be other objects moving in the front of the analysed object. In this case, the analysis starts all over again, this time however only the area occupied by the object previously analysed is taken into account. The above process is repeated recursively until the scene decomposition is satisfactory (Figure 2.17). Once all the objects are separated, their shapes are encoded. Independent shape extraction techniques are presented in [diehl91] and [wang94]. The encoded shapes of all the objects form the M-set.

Encoding motion (the A-set) is the next step of the analysis. Again, the motion in the frame is considered to be the superposition of the motion of individual blocks within the frame. If a rigid source model is used, each separated object can be mapped onto the 2D space by two co-ordinates. It has been shown in [tsai81], that those co-ordinates can be described by 8 a-coefficients arranged in following form (2.6).
The \( a_1-a_8 \) constants denote coefficients describing movement, \( X' \) and \( Y' \) are the co-ordinates of the former object location, \( X \) and \( Y \) are the current location co-ordinates.

\[
\begin{align*}
X' &= a_1X + a_2Y + a_3 \\
Y' &= a_4X + a_5Y + a_6 \\
&= a_7X + a_8Y + 1
\end{align*}
\]  \hspace{1cm} (2.6)

Motion can be described as the difference between the co-ordinates of the object in subsequent frames. The \( a \)-coefficients are normalised, quantised and transmitted. Detailed description of the motion of a single object in the scene requires sequential application of (2.6).

Coding of colour information (S-set: luminance and chrominance) is the last step in the sequence analysis. According to [musmann89] a sequence coded using the OBASC technique looks better than the same sequence coded using a block-based technique even in the presence of a bigger mean square error. The geometrical errors introduced by OBASC are claimed to be ‘less visible’ and ‘less annoying’ than errors introduced by too heavy quantisation. It is also stated [hotter94] that, assuming the same bit-rate, ten times more data can be allocated for colour coding in an OBASC encoder than in a block based encoder.
Chapter 2: Moving image compression systems

The entire concept of OBASC has been followed quite enthusiastically by numerous contributors. In [wollborn94], a method of predicting the texture of the analysed object is described. In [ostermann94], a different source model (based on moving rigid 3D objects) was proposed and tested. Yet another source model has been described in [hotter91] - this time based on deformable 2D segments. A coder capable of switching between H.261 and segmented-oriented modes was described in [chowdhury94] and [chowdhury94-2]. The tests have shown that the object-assisted coding was helpful in coding of simple objects performing large motion (translational or rotational). Other issues concerning segment coding were visited by [martinez97] and [mulroy94].

As it has been proven in [hotter94], the object-oriented moving image coding technique (OBASC) is capable of creating better results than block-based techniques. However in common with the waveform based techniques, the OBASC bit rate is too high for application to extremely low bit rates. It is, however, clear that this type of technique would be more relevant to compression of video signal to data rates of below 10 kbit/s. This is because, due to the nature of the encoding system, there are no blocky artefacts. Indeed, efforts have been made to merge object-oriented and semantic based approaches [kampmann97, kampmann97-2, kampmann97-3]. They have resulted in the creation of a system called KBASC (Knowledge-Based Analysis-Synthesis Coding).

2.2.2.2 Semantic model or knowledge based coding

The major difference between object-oriented and semantic model (or knowledge) based coding is that the latter makes certain assumptions as to the contents of the scene. If the scene contents are roughly known, an optimised coding system may be applied. This approach is particularly relevant in the case of head-and-shoulders and head-only scenes. The content may be modelled by a pre-defined wire-frame, thus reducing the data-rate very considerably. Since the author believes that there is a strong connection between semantic analysis in head-and-shoulders scenes and facial recognition, the two problems will be investigated together in a separate chapter.

In the field of semantic model based coding, techniques seeking a solution to the so-called 'videophone problem' (i.e. sending a moving image in real time via a standard PSTN telephone link) has attracted most attention. In the case of a typical videophone scene it can be assumed that the face of the speaker is visible at most times. The most commonly-used
model of a head-and-shoulders scene is that of *Candide* [rydfalk87, forchheimer84] - a wire-frame consisting of triangles only Figure 2.18

![Candide wire-frame](image)

Figure 2.18: *Candide* wire-frame

Although more detailed wire-frames have been proposed [parke74], [welsh90](Figure 2.19), [aizawa89] it seems that *Candide* in both its forms presents a good trade-off between the complicity and fidelity of reconstruction. *Candide* is also the most commonly used wire-frame model of the human face.

![Candide -2 wire-frame](image)

Figure 2.19: *Candide* -2 wire-frame

The wire-frames mentioned above are also referred to as ‘geometric models’. This type of wire-frame does not have any physical meaning. On the other hand, ‘physical facial models’ are built based on the anatomy of human face (i.e. they take into account the layout of facial muscles, etc.). Although more detailed, the physical models also more difficult to analyse [terzopoulos93, lei96]. Lack of physical context in geometric models (e.g. *Candide*) is compensated for by inclusion of action units [ekman77]. An action unit can be regarded as a
quantised facial expression. It describes the motion of certain group of vertices (e.g. left eye, lips - Figure 2.20). The issue of action units will be presented in more detail in later sections of this thesis.

Figure 2.20: Eyes’ and lips’ wire-frame model details

The semantic based coding of a typical videophone sequence can be described in the following way (Figure 2.21). Since the approximate content of the scene is known, both
transmitter and receiver share the same model of the scene (although this is not a
prerequisite, e.g. a model of the scene can be sent as a header during the communication
process). Once the connection is established, the encoder would send the texture of the first
frame of the sequence. At the same time the encoder would have to adapt a generic Candide
model to the face in the actual scene. The adaptation process (also referred to as fitting) must
be carried out automatically. Once the scene model is fitted, the receiver must be able to
automatically track the face of the speaker at the transmitter side. The motion of the speaker’s
face would be transformed (again - in real time) into motion of the scene model (wire-frame).
The 3D co-ordinates of the model would then be sent to the receiver instead of the texture of
the analysed frame of the sequence. The task of the receiver would be to reconstruct the new
image based on the texture of the initial frame and the new co-ordinates of the scene model
using computer graphics techniques (texture mapping).

As can be clearly seen, successful implementation of the above codec would enable huge
savings of bandwidth, since it would be sufficient to send the updated 3D co-ordinates of the
analysed scene, rather than the texture of every frame of the sequence. The technique
described above was first proposed by [aizawa89] and [forchheimer89]. It was estimated that
the bitrates may be as low as 500-1000 bit/s, even without elimination of statistical
redundancies in the bitstream of 3D coefficients.

The early ideas that finally led to formulation of the wire-frame based coding system can be
found in the work of two independent research groups [forchheimer83] (University of
Linköping) and [parke82] (Parke’s group). Since that time there have been numerous
proposals concerning semantic model-based coding systems. Due to the complexity of the
problem, a coherent codec has not yet been built. However, since the knowledge based
approach is the only technique offering compression ratios suitable for extremely low data-
rates, it deserves further consideration. This is the topic of the following chapter where the
main problems facing knowledge (semantic) based moving image coding are identified and
solutions are proposed.

2.2.3 MPEG-4 and layered coders

The wide range of proposed moving image encoding systems has resulted in a change of
direction in the work of the MPEG-4 standards group. At the present moment, one of the
main concerns of MPEG-4 is the description of a universal codec capable of encoding a
Chapter 2: Moving image compression systems

bitstream for all ranges of data-rates (MPEG-4 is also designed to handle other types of multimedia information). This type of encoder may be referred to as 'layered coder' [musmann95] (switched coder [chowdhury94]). As there seems to be no universal solution for all data-rates, the layered coder would have to use other techniques in order to achieve the task of multi-rate encoding capability. The structure presented in [musmann95] consists of five layers each analysing the incoming video signal in parallel (Figure 2.22). The outputs from all layers are fed into the layer selector. The task of the selector is to choose the layer that encodes the image with the highest efficiency (for a given bandwidth).

![Layered video coder diagram](image)

Figure 2.22: Layered video coder

The five layers cover a range of different capabilities. Layer I coders are not capable of encoding any motion information, i.e. only intraframe (probably block or region based) encoding is involved. Most of the currently used techniques would fall into the category of layer II encoders. MPEG, MPEG-II, H.263 have motion encoding capabilities and it is also possible to apply these techniques to intra-frame encoders only. A layer III capable encoder, in addition to being capable of encoding colour and deriving motion vectors, must have a mechanism for encoding parameters describing the shape of the objects in the scene. However, in case the objects cannot be properly segmented, the layer III encoder must have the capability to switch back to layer II or even layer I mode (the same is true for layer II encoders: they must be capable of switching to layer I). OBASC-like encoders [musmann89]
can be treated as layer III encoders. Layer IV and layer V encoders both use a priori knowledge about the contents of the scene. It is suggested in [musmann95], that layer V encoders should additionally utilise the concept of action units in order to increase compression ratio, should this prove necessary.

What is worthy of note, is that three out of five layers in the proposed coder incorporate techniques which can be described as model based. Furthermore, two out of three of the model based techniques are in fact knowledge based. This indicates the importance of model based, and knowledge based techniques in particular, in future research.

Although MPEG-4 very much concerns moving image coding techniques, no particular technique is specified [chiariglione97]. Perhaps the fact that the standard is open for new types of data and new algorithms is its greatest strength. Each type of information (video, sound, text, still image, synthetic data, etc.), referred to as an ‘AV object’ (Audio-Video object), is treated on an equal basis. At the transmitter side, the AV objects are encoded (along with their temporal relationships, if any), multiplexed and sent through the transmission channel. At the receiver side, those components are de-multiplexed and decoded. This scenario is reminiscent of well-established video coding standards such as MPEG and MPEG-2. However, in case of the MPEG-4 standard the type of data is not limited to video, sound and their temporal relationships, which is the case in both MPEG and MPEG-2 standards.

Another important difference between MPEG-4 and earlier MPEG standards is the extensive transmitter-receiver interaction. The introduction of this feature was inevitable because of the openness of the entire system. Although most types of AV objects will be known to the transmitter and receiver a priori (e.g.: video, audio), the receiver will be able to acquire information on the structure of the new data type. What is even more significant is that the receiver will be able to download the tools it needs to decode the new data type. Obviously, it will also be possible to download new tools for the decoding of ‘traditional’ AV objects if the receiver cannot understand the encoding ‘language’ (i.e. if the encoding standard is not known a priori). It would be reasonable to expect, that the standard ‘languages’ will include MPEG and MPEG-2, i.e. that MPEG-4 would be downward compatible with its predecessors. The openness of the architecture is reminiscent of the protocols used by the Internet suggesting that the relevance of the entire MPEG-4 standard may become questionable. However, the MPEG-4 standard is not only about transmission protocols (i.e. the system
layer - also referred to as MSDL - MPEG-4 Systems Description Language). As was mentioned before, MPEG-4 is open for new tools (software) and new data-types (i.e. synthetic). Although at the present time it is premature to speculate which software tools will be known \textit{a priori} it is has already been stated, that apart from the creation of a universal 'multimedia transmission language' the MPEG-4 standard targets data-rates between 5-64 kbit/s for mobile applications and up to 2 Mb/s for TV/Film applications \cite{sikora97}.

<table>
<thead>
<tr>
<th>FUNCTIONALITY</th>
<th>MPEG-4 TASK</th>
</tr>
</thead>
<tbody>
<tr>
<td>CONTENT-BASED INTERACTIVITY</td>
<td></td>
</tr>
<tr>
<td>'Content-Based Manipulation</td>
<td>Content-based manipulation and bitstream editing</td>
</tr>
<tr>
<td>and Bitstream Editing'</td>
<td>without the need for transcoding</td>
</tr>
<tr>
<td>'Hybrid Natural and Synthetic Data</td>
<td>Combining synthetic scenes or objects with ordinary video, interactivity</td>
</tr>
<tr>
<td>Coding'</td>
<td></td>
</tr>
<tr>
<td>'Improved temporal Random Access'</td>
<td>Random access (within limited time and with fine resolution) to individual</td>
</tr>
<tr>
<td></td>
<td>frames and objects on the video scene</td>
</tr>
<tr>
<td>COMPRESSION</td>
<td></td>
</tr>
<tr>
<td>'Improved Coding Efficiency'</td>
<td>Subjectively better visual quality at the same bit-rates</td>
</tr>
<tr>
<td></td>
<td>compared to existing or emerging video coding standards</td>
</tr>
<tr>
<td>'Coding of Multiple Concurrent Data</td>
<td>Coding of multiple views of scene efficiently. For</td>
</tr>
<tr>
<td>Streams'</td>
<td>stereoscopic video applications - ability to exploit</td>
</tr>
<tr>
<td></td>
<td>redundancies in multiple viewing points of the same</td>
</tr>
<tr>
<td></td>
<td>scene, compatibility with normal video</td>
</tr>
<tr>
<td>UNIVERSAL ACCESS</td>
<td></td>
</tr>
<tr>
<td>'Robustness in Error-Prone Environments'</td>
<td>Error-robustness to allow transmission over variety of</td>
</tr>
<tr>
<td></td>
<td>wireless and wired networks and storage on various media.</td>
</tr>
<tr>
<td></td>
<td>Sufficient error robustness for low bit-rate</td>
</tr>
<tr>
<td></td>
<td>applications under severe error conditions (e.g. long error bursts)</td>
</tr>
<tr>
<td>'Content-Based Scalability'</td>
<td>Ability to achieve scalability with fine granularity in</td>
</tr>
<tr>
<td></td>
<td>content, quality (e.g. spatial and temporal resolution),</td>
</tr>
<tr>
<td></td>
<td>and complexity.</td>
</tr>
</tbody>
</table>

Table 2.5: Requirements of MPEG-4
With respect to the existing video standards, MPEG-4 introduces seven new functionalities grouped in three sets: content-based interactivity, compression and universal access (Table 2.5).

Future multimedia applications are supposed to be more 'interactive' and 'user friendly'. Encoding the individual elements of the real-life video scene as separate objects may give the concept of 'interactivity' a new meaning. Each scene of the video sequence could be manipulated by the user. If, for example the original scene contained two speakers, then the user could demand one of the speakers to be removed from the scene along with the sound track assigned to that speaker. The spatial orientation of each speaker could be changed. In other words, the scene viewed by the user could be very different from the one seen by the encoding device. All the differences would be created on demand of the user (receiver), while the transmitter would provide an entire range of options for manipulating the objects in the scene. The bit-stream created in the transmitter would be 'object-layered'. The shape, transparency, spatial co-ordinates, motion restraints, etc. would be described as a part of the bitstream describing a particular object. The receiver has then an option of displaying all of the encoded objects or only a certain sub-set. Additionally, the scene 'reconstructed' in the receiver can contain new elements - never 'seen' by the encoder. All these functionalities are envisaged to be available without re-encoding of the bitstream. The above description summarises the concept of content-based interactivity [pereira97, sikora97].

In its pursuit of increased compression without degradation of the image quality at all data-rates, the MPEG-4 group admits, that 'improved coding efficiency, in particular at very low bit rates below 64 kbit/s, continues to be an important functionality to be supported by the standard' [chiariglione97], including 'wireless communications and database access'. This is where the concepts of universal access and compression meet: MPEG-4 targets all future means of communication: including very noisy or very low bit-rate channels.

Anticipating rapid development of general purpose DSP hardware, the MPEG-4 group tried to create as few constraints as possible. MPEG-4 will therefore enable mechanisms to download missing software decoder tools at the receiver. At this stage it is worth mentioning and explaining the idea of MPEG-4 'tools', 'algorithms' and 'profiles'. An MPEG-4 tool is an expression used for describing abstracts such as object description, motion compensation etc. The MPEG-4 tools constitute the MPEG-4 'toolbox' (Figure 2.23). MPEG-4 algorithms would then allow the combination of the tools into algorithms that would be used to display
natural or synthetic video scenes. The MPEG-4 profile defines a sub-set of algorithms and tools that would be used by the decoding device depending on the channel bandwidth, error resilience, degree of complexity of the decoder, minimal decoding delay etc.

If a certain ‘tool’ is not present in the decoder, it can demand its download from the encoder (Figure 2.24). It is worth noticing that the toolbox must include tools for decoding both the natural and synthetic data. This might allow combining the two types at the receiver side, i.e. construction of a hybrid natural-synthetic decoder. The last statement is particularly important for semantic model-based moving image coding techniques, since it confirms inclusion of such techniques in the envisaged standards. Moreover, MPEG-4 will also allow a mixture of pixel-based (waveform based) and model-based (including semantic-based) techniques. This approach is consistent with (or maybe inherent to) the theses included in [aizawa89].

The concept of ‘video object planes’ (VOPs) is one of the most important new abstracts introduced by the MPEG-4 group (as far as video coding is concerned). Each frame of the video sequence can be viewed as a group of VOPs. Therefore, each object constitutes a different VOP. This is not to say, that, e.g. in the case of a head-and-shoulders scene a VOP is the head of the speaker, or the eye of the speaker. A VOP can be equivalent to the actual object on the scene, but it does not have to be. The segmentation of a video scene into a VOP can be performed either on a pixel-by-pixel or motion homogeneity basis. Therefore in some cases the VOP may correspond with actual objects in the scene. If so, the scheme is reminiscent of that proposed in [mussman95]. If not, i.e. when the segmentation is carried out basing on texture homogeneity, rather than motion homogeneity, we are dealing with segmented image coding [cortez95, gilge89, brigger95, zhang95].
A VOP can be shaped arbitrarily. It is the task of the encoder to include its shape in the transmitted signal. Also, the shape of VOPs will vary from frame to frame. Successive VOPs belonging to the same physical object in a scene are referred to as 'video objects' (VOs). The information about the shape, texture and motion of all VOPs belonging to a particular VO are encoded into a 'video object layer' (VOL). VOL would also include information about spatial position of each VOP. As a result of the above arrangement, each VOL can be encoded independently. Although encoding is independent for each VOL, MPEG-4 uses an identical algorithm to code shape, motion and texture in each of the object layers.

The idea of VOPs can be explained on a simple example of a head-and-shoulders scene (Figure 2.25). The image is decomposed into two VOPs (VOP1 - stationary background and VOP2 - the speaker). In this case each VOP will constitute a separate VOL, since background and speaker can be viewed as separate objects. Although it can be seen that both VOPs are non-overlapping, they do not have to be in the general case. It is possible that the contents of the entire background (without the speaker obscuring it) are known prior to the transmission. In this case VOP1 would be rectangular and VOP2 - arbitrarily shaped.
Figure 2.25: Typical 'head-and-shoulders' scene subdivided into two independent VOPs

Figure 2.26: Standard and contour macroblocks in MPEG-4
If the original sequence is not subdivided into individual VOLs, the system ‘degenerates’ to a standard block-based coding scheme (MPEG, MPEG-2, H.263). Because of that, MPEG-4 is often referred to as a ‘logical extension’ of previous MPEG standards. If the analysed video sequence consists of rectangular frames only, the shape information is not encoded into VOL, and the encoding algorithm is ‘similar’ to those presented in the MPEG, MPEG-2 and H.263 standards.

There are numerous similarities between the coding of arbitrarily shaped regions in MPEG-4 and the rectangular blocks in the existing block-based algorithms. MPEG-4 introduces the concept of I-VOP (intraframe video object plane), P-VOP (predicted video object plane) and B-VOP (bi-directionally predicted video object plane). The definitions are similar to those used in MPEG standard (I, B, P frames - Figure 2.2). In the case of I-VOPs, only the information from the current frame is used in the encoding process. Encoding of the P-VOP planes uses data from the nearest previously encoded VOP. Similarly, in the encoding of the B-VOP planes both information from nearest previous and subsequent VOP is used. Motion compensation is carried out similarly to that described in the MPEG and H.263 standards. Here, two distinct types of macroblocks are used: ‘standard macroblock’ (or macroblock in MPEG sense) and ‘contour macroblock’ (Figure 2.26).

Although MPEG-4 acknowledges that block-based techniques create very disturbing artefacts, a square grid is still used for encoding of individual VOPs. In MPEG-4 the macroblock grid is used for motion estimation, motion compensation and coding of the texture of the object. The entire VOP is padded by padding the contour macroblocks. Motion compensation of the standard macroblock is similar to that used in earlier standards. In the
case of contour macroblocks, a polygon matching (instead of block matching) technique is used. In the polygon matching technique, the pixels not belonging to the original (not padded) VOL are not taken into account during motion estimation. Similarly to MPEG, the texture is encoded after the macroblock is split into 4, 8 x 8 blocks of luminance and two blocks of chrominance of the same size. Again, special arrangements are necessary for blocks from the contour macroblocks. For the purpose of encoding of these blocks, the image is padded with pixels set to level 128 (mid-grey), Figure 2.27. Standard block-based DCT is then performed. This is followed by scanning and quantisation of DCT coefficients. Run-length encoding using the tables borrowed from MPEG/MPEG-2 and H.263 standards is performed in the final step.

At the present time, all tools defined in the MPEG and H.263 standards (DCT, motion estimation, motion compensation) are supported by MPEG-4. MPEG-4 also supports most of the tools defined in the main profile of MPEG-2. It therefore ensures that performance of the MPEG-4 standard at very low bit rates is at least identical to the H.263 standard. In the proceedings of the MPEG-4 standards it is also mentioned that it can be made almost compatible with MPEG and H.263 standards. This implies that the MPEG-4 standard is not entirely compatible with its predecessors.

2.3 Summary

The waveform-based moving image coding techniques (block DCT transform, VQ, fractal, wavelet) seem to perform well at data rates higher than 64 kbit/s. Although several standards are concerned with encoding video for data rates lower than this limit, it seems that the lower bit rate boundary for waveform based techniques has been reached, and further compression can only be achieved at the cost of serious degradation of picture quality.

Model based techniques (object oriented and knowledge based) are a relatively young branch of moving image coding techniques but seem to be the only technique offering a viable alternative to 'well established' waveform based algorithms. Although the object oriented techniques offer better picture quality at very low data rates, their compression ratios are comparable to those offered by traditional (waveform) approaches. On the other hand, semantic model based techniques offer the potential for compression ratios absolutely unattainable by the conventional algorithms. Although there are numerous unresolved problems concerning knowledge based coding, these techniques will have to be applied, if
compression of video is ever to be realised at extremely low data-rates. This thesis proposes a solution to some of the most difficult problems in semantic model based coding: semantic wire-frame fitting (adaptation) and semantic wire-frame tracking. These two issues will be described in more detail in the later chapters.
3.1 Introduction

Since (as shown in the previous chapter) a knowledge based approach to moving image coding seems to be the only option for extremely low data rate systems, this chapter will look in more detail at the problems that need to be solved before a complete knowledge based codec can be built.

Since the author believes that in typical videophone scenes (head-and-shoulders, head-only) - semantic based techniques and facial recognition systems have a lot in common (Figure 3.1), a survey of the most promising face recognition algorithms will be presented.

Figure 3.1: Knowledge based coding and face recognition for analysis of a videophone scene

Face recognition systems are mainly used in law enforcement, video surveillance, credit card identification, etc. This thesis presents an entirely new approach which merges facial recognition and moving image compression techniques. Although humans have no problems
recognising faces even in cluttered backgrounds, automatic (machine) face recognition appears to be a very difficult task. From a statistical point of view, faces are just another waveform. The symmetry of a face - obvious to any viewer, is not that obvious to the machine. For the machine, an image of a face is just another waveform to be analysed and its symmetry can be easily destroyed e.g. by using inappropriate lightning conditions. In the area of face recognition the past decade has been relatively fruitful. This is mainly due to the growing power of widely available computing facilities. Historically, face recognition systems have always been computationally intensive, so appropriate hardware has always been a necessary prerequisite in this area of research. The main problem with assessing the accuracy of face recognition systems is that in most cases they are tested on a certain custom built image data-base. In this situation, a reliable recognition technique may be identified by looking at the size of the database it is able to operate on and still give positive results. In other words a face recognition system giving 80 % correct results using a data-base of 1000 images is probably more worthy of attention than a system giving 100 % of correct results from a data-base of 100 images, even if the images seem to be more complicated (e.g. to have cluttered backgrounds).

Semantic-based coding techniques require the accurate detection and tracking of a human head/face and the fundamental facial features (eyes, lips) in a scene. There are numerous existing techniques targeting the automatic location of human faces and fundamental facial features in a scene. Videophone scene contents analysis is usually performed in a top-to-bottom fashion (Figure 3.1). In the first step the silhouette of the head of the subject (speaker) must be detected. This is followed by detection of the boundary of the face. The final stage involves a search for the fundamental facial features. The analysis of the contents of the scene is closely related to the analysis of the motion in the scene (Figure 3.1). Semantic-based moving image coding and face recognition share very similar problems and can utilise very similar techniques for successful solution of these problems.

This chapter is organised in two main subsections. One concerns problems in knowledge based coding and the other tries to discover if these problems can be solved with the help of face recognition algorithms. As will become apparent, the problems with wire-frame fitting and tracking are very similar to those in the extraction of speaker's silhouette, face, etc. (which is a part of the face recognition process). The mathematical tools used for facial recognition will also be examined and evaluated for their suitability.
3.2 Main problems of knowledge based coding

Although the concept of knowledge based techniques is relatively straightforward, the actual (software) implementation of the codec is quite a challenging task. The generic wire-frame must first be accommodated (fitted) to the actual scene. Once fitted, the wire-frame coordinates must be updated on a frame-by-frame basis. Fitting and tracking is particularly important in the areas occupied by the facial features. Slight misjudgement might lead to disturbing artefacts (e.g. vertices representing corner of the lips in the wire-frame must be fitted perfectly to the corner of the lips of the speaker). Although tracking might be viewed as a frame-by-frame fitting, it is more practical to view the two problems separately. A properly fitted model may be tracked more effectively (as far as accuracy and speed is concerned) using a dedicated technique. Following the discussion of fitting and tracking there is a subsection describing other problems relevant to the area of knowledge based techniques.

3.2.1 Wire-frame fitting (adaptation)

One of the fundamental contributions on model based coding [aizawa89] assumes ‘semi-automatic’ fitting of the wire-frame. Obviously in real-life applications such an approach is not acceptable. This contribution is nevertheless worth looking at since it was the first time texture mapping techniques were proposed and issues concerning the accuracy of model fitting are emphasised. First, it is necessary to re-scale the generic wire-frame. For this purpose four characteristic points are manually extracted form the object’s face (Figure 3.2). This adjusts the wire-frame dimensions and roughly fits it to the face.

Figure 3.2: ‘Rough fit’ (Aizawa)
Subsequently all the edge vertices (vertices describing edges of the face and hair line) that are not in place as a result of the rough fit are back-projected onto the edges of the face and the hair line respectively. As reported in [aizawa89] this rough fit alone already produces quite good results when global motion (3D motion of entire model) is applied. However, in order to fit the wire-frame accurately, the vertices describing the facial features must be put in place (a rough fit rarely puts the facial features’ vertices in place since models are prepared based on generalised information about the human face). An accurate fit of the wire-frame in the areas shown in Figure 3.3 was judged to be sufficient.

Once the vertices of the wire-frame are fitted both globally (edges of face and hair line) and locally (facial features) the entire model can be animated using global motion vectors (rotation, translation) and action units (responsible for motion of facial features, Figure 2.20) to create a life-like impression.

<table>
<thead>
<tr>
<th>ACTION UNIT (AU) NUMBER</th>
<th>ACTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>Raises upper lip</td>
</tr>
<tr>
<td>12</td>
<td>Pulls lip corners</td>
</tr>
<tr>
<td>15</td>
<td>Depresses lip corners</td>
</tr>
<tr>
<td>18</td>
<td>Puckers lips</td>
</tr>
<tr>
<td>20</td>
<td>Stretches lips</td>
</tr>
<tr>
<td>23</td>
<td>Tightens lips</td>
</tr>
<tr>
<td>25</td>
<td>Parts lips</td>
</tr>
</tbody>
</table>

Table 3.1: Selected action units (AUs)
There are 44 action units (AUs) described in the original facial action coding system (FACS) [ekman77]. Several action units can describe the motion of one facial feature. For example, lip motion is described by the action units from Table 3.1. Action units are responsible for motion of certain group of vertices at the same time as shown in Figure 3.4.

A very interesting approach was proposed in [seferidis91]. Here the original image was pre-processed in order to create a binary image containing the most significant features (edges). The problem of feature extraction was therefore reduced to the problem of finding of edges. The pre-processing was carried out using an edge detector based on mathematical
morphology. The morphological filters tend to simplify the image data while preserving the information about the shape of the image. The proposed edge detector is a simple linear combination of morphological filters. The image is processed in two passes: in the first pass a morphological erosion filter is applied to the original image. In the second pass a morphological dilation filter is applied to the same original image Figure 3.5.

![Mathematical morphology based edge detector](image)

The two resulting pictures are then subtracted and the thresholded difference forms the final image. The main advantage of this filtering method is that it is very fast - much faster than "traditional" edge detectors based on analysis of the gradient (a selection of which may be found in [jain89], [hall79], [lewis90] or [jensen93]). As a result of the application of the morphological edge detector and thresholding the image is converted into a two-level picture containing the edges of the facial features and the contour of the head. It is admitted in [seferidis91] that occlusion of, e.g. an eyebrow by hair causes mistakes in the location of that facial feature. It should also be noted however, that this method uses a single frame (not a sequence of frames or a two-frame view) to estimate the shape and position of the facial feature. The idea of pre-processing the source image (sequence of images) and transforming them into a bi-level representations seems to be the right way to go. The bi-level image is fairly 'similar' to the wire-frame model and it is easier to fit the model to a thresholded image than to a grey-scale one. In this case, the facial features extraction problem can be reduced to the problem of finding a fast and efficient edge detector. Once the edges of the face and
facial features are extracted, they need to be fitted to the actual wire-frame. A simplified Candide wire-frame contour was used here (Figure 3.6).

In the first step the wire-frame was re-dimensioned to the size of the speaker’s head in the Miss America sequence [seferidis91]. Since at this stage the wire-frame contour can be viewed as a binary picture, the image representing the edges of the subject can be matched to the contour of the model using a dedicated binary matching technique. A technique called ‘chamfering’ was used [barrow77]. Although the proposed algorithm has been tested on the Miss America sequence only, it is interesting due to its simplicity and low computing power requirements.

In [reinders93], the contours of the face are extracted manually. Then the contour image is used as a base image for general model adjustment. A set of six points is used in order to simplify the analysis. They are placed in the corners of the eyes, lips and on the intersection of a vertical line splitting the face image into two symmetrical parts and a horizontal line connecting the points in corners of the eye. Again, these points are found manually. Having located these points the rotation of the head can be estimated by measuring the angles between the lines connecting them. After the global motion was approximated (i.e. after the angles were found) the general model of the face was superimposed onto the contours of the face. The proportions of the model are subsequently adjusted to those of the contour image. As a result of that the general face model covers the contour of the face and also some areas outside the face contour. Therefore in the next step the contour of a general model of a face
Chapter 3: Knowledge based coding and facial recognition

(re-sized by now) must be back-projected onto the face contour. The final step of the procedure is to adjust placement of the vertices of the wire-frame model. This task is simpler and in most cases adjustments are minimal.

Another method of adaptation of a generic 3D model of the human face was proposed in [reinders95]. First, the silhouette of the speaker is isolated from the background. In order to achieve this a spatio-temporal approach is adapted. First the difference between subsequent frames is low-pass filtered (in a spatial filter). Subsequently several frame differences are fused together in order to obtain a more pronounced silhouette of a moving subject (head-and-shoulders type speaker). As a result of this processing several candidate regions are obtained. The biggest candidate region is chosen as the one to be the silhouette of the speaker. Once the silhouette is extracted, the algorithm attempts to localise the speaker’s head. This is done based on an assumption that there is a pair of concavities separating the head from the shoulders. The concavities are searched from the top to the bottom of the silhouette along the vertical axis starting at the highest placed pixel of the silhouette. The sum of pixels on the left and on the right of the vertical axis gives the approximate thickness of the silhouette at a given y co-ordinate. Concavities are located when the global minimum in this top to bottom search is found.

Very recent contributions is the area of automatic semantic wire-frame adaptation are those trying to fuse the semantic and object oriented approaches. The KBASC system [kampmann97] (mentioned earlier) was proposed in as an extension of OBASC (object-based analysis-synthesis codec) [musmann89]. KBASC (knowledge-based analysis synthesis codec) differs significantly from the OBASC in the sense that it makes extensive use of pattern recognition techniques. However, it cannot be listed as purely semantic-based technique, since the system uses the knowledge about the structure of the scene for different purposes. The OBASC system encodes each moving object by three sets of parameters and it does not have any mechanism for analysing the scene contents. KBASC is essentially a ‘two layer’ [musmann95] encoder with the object oriented approach being the main and fail-safe encoder in case semantic encoding (again using the Candide model) was not possible. For the purpose of automatic wire-frame fitting, the algorithm recovers the positions of the eyes and the lips. The recovery is carried out based on an assumption that a speaking person moves its eyes and mouth in addition to its head. Furthermore, certain assumptions as to the geometry of the human face are taken into account (i.e. that the eyes and the lips span an isosceles triangle). Initially, the coder has no knowledge about the contents of the sequence and the entire scene
is modelled by a generalised 3D model. Once the silhouette of the moving object roughly approximates a head-and-shoulders, the encoder assumes that there may be a person in the scene. It isolates the position of the head and performs a search for facial features (the eyes and the lips). The eyes are searched for in the upper part of the area potentially occupied by the head, the lips are searched for in the lower part of this area. In order to find the features, the algorithm uses correlation between the subsequent frames in the sequence. It also utilises template matching (with templates representing lips and the eyes of the same subject) and horizontal edge detectors). As reported by the authors incorporation of the face model into the object-oriented encoder allows a reduction of the number of bits necessary to encode a sequence. Although the reduction is substantial, the ultimate bit rate lies well above the 10 kbit/s rate. This is because once the wire-frame is fitted, the system encodes the entire area of the silhouette using techniques described earlier for OBASC [musmann89].

In general there is a plethora of feature extraction systems leading to automatic wire-frame fitting. The techniques presented in this chapter at least report the fact that they had been tested on a certain set of images or on (at least one) image sequence. Other techniques will not be described here in more detail, since they do not seem to have been tested on reasonable range of subjects or sequences. The general impression after reviewing the papers concerning automatic adaptation of a wire-frame is that all the techniques make too general assumptions about the geometry of the human face. There are lots of heuristic clauses and ‘if’ statements. A face, unfortunately, must be viewed as an ordinary waveform. In this case, it seems that the method of wire-frame fitting should use more comparative information (e.g. information about other faces) rather than geometrical restraints.

3.2.2 Wire-frame tracking

The first complete approach to tracking problems in model-based coding was presented in [li93, li94-2]. Here optical flow analysis was used. Optical flow is a mathematical model used to estimate movement in the scene [horn86]. The form of optical flow depends upon constraints that were accepted to construct it. Once all the constraints are accepted, the task of optical flow is to derive equations for a optical flow field. The optical flow field is a two-dimensional vector \((u, v)\) describing movement of each point on the scene surface (3.1).

\[
\begin{align*}
  u &= \frac{dX}{dt} \\
  v &= \frac{dY}{dt}
\end{align*}
\]  
(3.1)
where \( u \) and \( v \) denote vector co-ordinates and \( X \) and \( Y \) - co-ordinates of the surface of the image.

Optical flow can be interpreted as the speed of a pixel in the image. The speed of the pixel is dependent upon its placement on the surface of the image. Some additional constraints are required in order to solve (3.1). The analysis of optical flow can start on a two-frame basis ('two view' optical flow). The optical flow field can be global (describing global movement only, in the case of head-and-shoulders scenes global motion denotes motion of the wire-frame as a rigid body), local (where the vertices of the wire-frame are allowed to move with respect to each other, e.g. eye's open-close), or joint (describing both global and local movement by the same vector). Optical flow forms are also dependent upon the approximation techniques, that are used for estimating motion parameters.

Analysis of optical flow in the image is relatively expensive computationally. An optical flow model applied to a two-view estimation of a head-and-shoulders scene was proposed in [li94-2] and is described by the following set of equations. According to the presented analysis the motion of an arbitrary point \( s(x,y,z) \) in the image could be described by certain set of equations (3.2). The new co-ordinates of a point \( s \) after transformation \( (s'(x,y,z)) \) can be calculated as a superposition of its global and local motion (the \( x, y, z \) co-ordinates in 3D space should not be mistaken for \( X, Y \) co-ordinates in 2D space from (3.1)).

\[
\begin{align*}
    s' &= R \cdot s + T + E \cdot \Phi \\
    s &= (x, y, z)^T \\
    s' &= (x', y', z')^T \\
    R &= I + \begin{pmatrix}
        0 & \Omega_z & -\Omega_y \\
        -\Omega_y & 0 & \Omega_x \\
        \Omega_y & -\Omega_x & 0
    \end{pmatrix} \\
    T &= (T_x, T_y, T_z)^T \\
    \Phi &= (\phi_1, \phi_2, \ldots, \phi_m)^T \\
    E &= \begin{pmatrix}
        e_{11} & \ldots & e_{1m} \\
        e_{21} & \ldots & e_{2m} \\
        e_{31} & \ldots & e_{3m}
    \end{pmatrix}
\end{align*}
\]

The global motion is described by two matrices: \( R \) and \( T \). The local motion is described by the two remaining matrices \( (E, \Phi) \). The \( R \) matrix is responsible for head rotation (\( \Omega \))
represents angular velocity about the respective axis), and the $T$ matrix is responsible for head translation ($T$ represents linear velocity along the respective axis). The $\Phi$ matrix is the so-called deformation vector. The deformation vector contains facial expression movement parameters (in $m$ rows, 1 column). The $E$ matrix contains a set of action units.

Using all the relations expressed in (3.2), the new position of a point in space $s'(x',y',z')$ can be expressed in the following way.

$$
x' = x + \sum_{i=1}^{n} e_i \phi_i + \Omega_x y - \Omega_y z + T_x
$$

$$
y' = y + \sum_{i=1}^{n} e_i \phi_i - \Omega_x x + \Omega_y z + T_y
$$

$$
z' = z + \sum_{i=1}^{n} e_i \phi_i + \Omega_x x - \Omega_y y + T_z
$$

The above equations represent the pixel position in 3D space. Before applying them to (3.1) they must be converted into 2D formulas. In this example, it is assumed that the 3D model can be projected onto a 2D surface using perspective. In that case the link between 2D space and 3D space can be expressed as:

$$
X = f \frac{x}{z} \quad Y = f \frac{y}{z}
$$

Thus the optical field derivatives are:

$$
u = \frac{dX}{dt} = f \frac{dx}{dt} \frac{z - x}{z^2}
$$

$$
u = \frac{dY}{dt} = f \frac{dy}{dt} \frac{z - y}{z^2}
$$

One of the most difficult problems in the analysis of optical flow in the case of head-and-shoulders, or head-only scenes lies in the separation of pixels performing global motion (i.e. rotational or translational) only and pixels performing local motion only (i.e. facial features motion) so that (3.2) becomes applicable. One cannot be sure what pixel performs what kind of motion. It is necessary to derive a technique for separating the pixels, i.e. to derive reliable feature extraction technique before the optical flow field can be calculated. This requires accurate feature extraction techniques, which may prove computationally expensive. Another approach is to treat local motion as noise and add it to global motion. However, that would deform the global motion parameters even more and lead to unacceptable results.
As it can be seen, calculation of optical flow may prove very difficult and in some cases impossible without proper extraction techniques applied in advance. Since extraction of features in sequences like head-and-shoulders is equivalent to extraction of the 3D position of the model, the calculation of the optical field might prove redundant as a whole. It is sufficient to extract the location of certain points on the human face to know exactly what is the 3D position of the head in relation to the camera and thus to fit the wire-frame model successfully. The results delivered by the analysis of optical flow are generally rather disappointing and taking into account the computational costs, this approach is not attractive. It seems that a simpler approach (taking into account the fact that the analysed object is in fact a human face) might be possible.

A simple wire-frame model tracking system was proposed in a series of contributions ([clark92], [kokuer92], [kokuer94]). A rule based approach allowed the authors to track the motion of a head-and-shoulders subject and transform it directly to the motion of a 3D model. The image is treated with a Nagao (valley) [nagao72] edge detector (also used in the work of [welsh91] together with the concept of active contours - snakes) in order to produce a binary image. Once the head and facial features are found, the tracking process commences. In the second frame (i.e. in the frame that followed automatic fitting) the position of the head (and facial features) is assumed to be the same as in the previous frame. The search for the new position is performed in a spiral manner until cross-correlation peaks (between the current and previous frame) are found. An arbitrary threshold ensures, that once the correlation peak decreases below a certain value (lower threshold), the tracking stops and the model is re-fitted using the technique described earlier. Another threshold is specified to avoid the tracking being held up in a local correlation maximum (upper threshold). If a correlation peak lower than the upper threshold is found the search technique continues (again in spiral mode) to search for another correlation peak.

Other tracking systems (not directly associated with video coding) were also proposed (e.g. [xie95], [toyama95]). Similarly to the case of automatic wire-frame fitting, although the problems of automatic tracking have been studied by various research groups, there seem to be few results of tests of these methods on a wide range of head-and-shoulders or head-only video sequences. Again, the tendency to rely heavily on certain heuristics prevails and the techniques seem to contain many ‘additional’ clauses resulting from the fact, that the analysed signal is a human face (or head-and-shoulders subject). Although utilisation of the
semantic knowledge about the geometry of the human face has its advantages, this knowledge should not be misused in the sense that the success of the application of the technique depends entirely on parameters derived from facial measurements.

### 3.2.3 Other problems: images and speech

Although the 'fitting issue' and the 'tracking issue' seem to be the two most important ones that must be solved before a successful knowledge based codec can be built, it would be unfair to omit the problems of speech encoding. This issue, although it will not be developed beyond this subsection, must be mentioned, since an entire videophone scene (i.e. video AND speech) must be encoded simultaneously and compressed into a bitstream at a data rate lower than 10 kbit/s.

One of the most obvious ways to compress speech in a videophone sequence is to utilise voice information for driving the lips. The voice-lips system is the only one in a videophone scene in which audio and video channels interact with each other. In [morishima91] an apparently successful system was presented. The system had two variations: a text to image converter and a voice to image converter. In the text to image conversion system, a model developed in [aizawa89] was used. The lips of the model were driven by text information. The analysis was based on phonemes.

<table>
<thead>
<tr>
<th>Phoneme classes</th>
<th>Phoneme vectors</th>
</tr>
</thead>
<tbody>
<tr>
<td>[a]</td>
<td>top edge of upper lip</td>
</tr>
<tr>
<td>[I]</td>
<td>bottom edge of upper lip</td>
</tr>
<tr>
<td>[e]</td>
<td>top edge of lower lip</td>
</tr>
<tr>
<td>[o]</td>
<td>bottom edge of lower lip</td>
</tr>
<tr>
<td>[u]</td>
<td>chin</td>
</tr>
<tr>
<td>[p][b][m]</td>
<td>outer lip edge (h)</td>
</tr>
<tr>
<td>[ø][w]</td>
<td>inner lip edge (h)</td>
</tr>
<tr>
<td>[r]</td>
<td>lip edge (v)</td>
</tr>
<tr>
<td>[dz][ts][z][s]</td>
<td>nose</td>
</tr>
</tbody>
</table>

Table 3.2 Selected phoneme classes

Table 3.3 Phoneme vectors
Seventeen phoneme classes were isolated. Each phoneme class can be approximated by a single mouth shape. Some of the phoneme classes are described in Table 3.2 and Table 3.3.

Some phonemes do not represent any specific mouth shape. In this case the mouth shape is derived by interpolating previous and subsequent phonemes. Each phoneme class addresses a set of nine vectors (Table 3.3). The direction of the vector is pre-defined (horizontal or vertical). In this way the only value defined by the phoneme class is the absolute value of the vector. The experimental results are claimed to be very good and encouraging. The system based on speech recognition was much more complicated and did not produce effects comparable with the system based on text. Unfortunately the general conclusion appears to be that audio information cannot be analysed reliably and used for image animation.

3.2.4 Summary

The most serious problems in the knowledge based moving image coding of typical videophone scenes seem to be automatic fitting and tracking of semantic wire-frames. Once solved, they would also be a good basis for the solution of other problems normally associated with model based techniques [aizawa95]: model selection, camera position estimation, lip synchronisation, text to speech (image) conversion, speech (image) to text conversion and model selection. Indeed, some of these problems may become obsolete should a reliable fitting/tracking technique be found.

This thesis will therefore focus on a solution to the 'fitting problem' and the 'tracking problem' basing on the analysis of a generic data-base of facial images.

3.3 The use of face recognition techniques

Since at this point image analysis meets face recognition, it is necessary to take a close look at existing face recognition techniques. The first step in facial recognition usually involves extraction of the speaker's silhouette. This is followed by localisation of the speaker's face and facial features. The two most promising approaches are described in separate paragraphs. The most reliable mathematical model is selected in the final part of this sub-section.
3.3.1 Feature extraction algorithms

Separation of the speakers face from the background image is the first step in face recognition algorithms. The techniques reported to date deal with image sequences or still images. Finding the face in an image sequence is usually easier since the additional information in the form of the frame difference proves to be a very useful tool. However, the analysis of frame differences often fails to provide reliable information in cases of illumination changes or a moving camera [chellappa95]. In many cases intra-frame information only is used to locate face in a video sequence.

The basic task of background separation has been addressed in [lettera89]. Here, a contour technique was applied in order to detect the boundary of the moving object (speaker). Both inter- and intra- frame information is utilised. The technique applies various edge detection algorithms (Sobel, Laplacian) in order to enhance the image of the moving contour from the image created based on frame difference. A similar technique is used in the mentioned earlier OBASC [musmann89] segmented oriented techniques.

Yet another segmentation method targeting extraction of the human face from the video sequence was presented by [li92]. The algorithm is based on the observation that the head and shoulders of the speaker undergo two different motions. It is subsequently assumed, that the motion of both shoulders and head can be approximated by affine motion equations. Segmentation of the speaker silhouette into head and shoulders is performed by separation of the motion field of the head and motion field of the shoulders.

Locating head boundaries using snakes was first proposed in [waite90]. This was followed by application of snakes to extraction of the human face [huang92]. In the basic form, a snake [kass87] is an energy-minimising spline which is influenced by three factors: internal contour forces, image forces and external constraint forces. The contour is initially placed near the analysed feature of the image and then allowed to 'contract' towards the lowest energy given by the following equation (3.6):

$$E_{\text{snake}} = \int \left[ E_{\text{internal}} v(s) + E_{\text{image}} v(s) + E_{\text{constraint}} v(s) \right] ds$$

(3.6)

where $v(s)$ is a parametric representation of the snake: $v(s) = (x(s), y(s))$. The fact that there are numerous local minima makes the reliability of this method questionable. However it was
applied on several static pictures by [lam94] and [gunn94] and on a bigger facial data-base in [welsh91]. In the latter we also find the application of the same method for extracting facial features.

The most straightforward method of detection of a face in a still image is to try and match a parametrised oval or polygon to the edge map of the analysed image. This method is referred to as ‘template matching’, where the oval (or polygon) is the template. It has been used in numerous applications to date. In [eleftheriadis95] an attempt to improve on the quality of traditional block-based techniques is described. The proposed project is fully compatible with an existing H.261 codec. It regulates the quantisation step of the encoder according to the content of the scene. If, for example, it was possible to detect a face, or at least a certain facial feature the coder would reduce the quantisation step for that area thus improving encoding quality. This approach to scene analysis is also referred to as ‘model-assisted’ coding, i.e. although as it will be shown, in reality - no model is involved in coding. In fact the technique tries to find objects that can roughly be estimated by an ellipse (e.g. head of the speaker). In the first step the sequence is downsampled in the temporal domain (the input luminance signal rate is reduced from 30 to 5 fps). The second step involves low-pass filtering in the spatial domain. The frame size is also decimated (reduced) by factor of 8 to 45×40 pixels. These frames are then subjected to an edge detection process involving the application of a Sobel operator. Finally the resulting images are thresholded. The actual search for ellipse-like objects (or ellipse fitting) is therefore performed on binary data. At each point, ellipse templates of various sizes and aspect ratios (various tilts were not tested) are tried-out for fit. The results of test on three video sequences were presented. The reliability of the results may however be queried, since the sequences seem to have been custom made.

Human face detection in complex background scenes (e.g. TV speakers) is the subject of another contribution [yang93]. The proposed systems consists of three stages. In the first two stages mosaic images are used while the final stage utilises an improved edge detection scheme. Application of mosaic images (images at different resolutions) allows the detection of faces without a-priori knowledge about the approximate size of the face (Figure 3.7). Faces in 83 % of the tested images were correctly detected. Although the method promises robust operation for faces of various sizes (also the number of faces in the processed image is not assumed to be known in advance), it is accompanied by a number of heuristic ‘eye rules’, ‘nose rules’ and ‘mouth rules’ (e.g. ‘the difference between the average grey levels of the
centre part and the upper round part of the quartet is quite significant’). This does not allow an objective evaluation of the accuracy of the method.

In [govindaraju89] an algorithm for finding human faces in newspaper photographs is described. In this method, the approximate size of the face must be known a-priori. Also the expected number of faces is required in advance. All the faces appearing in the photograph must be of approximately the same size. There are also certain additional constrains imposed on the contents of the analysed image. Here are some of them:

- The photograph must provide a front view of the faces.
- The faces must be upright, with almost negligible tilt.
- Faces should not be occluded by other objects.

The algorithm attempts finding a human face ‘by parts’: each part of the face silhouette (Figure 3.8) is extracted separately. In order to detect arcs, the generalised Hough transform is used [ballard81]. A standard Hough transform [Hough62] is used for detection of straight
The detected features are subsequently grouped using the semantic knowledge about the geometry of the human face.

One of the first systems for facial feature extraction was proposed in [fischler73]. Facial features were connected by a set of 'strings' (Figure 3.9).

![String model of a face](image)

Figure 3.9: String model of a face

This approach defines a local measure of how strong the particular feature is 'attracted' to a particular 2D position on the image. The measure for each element of the face model (Figure 3.9) is different. For example for the left edge of the face this measure would be the difference between the sum of intensity values on the left and on the right of the edge. It can be imagined that the above measure would produce large absolute values for the case when the left side of the edge contains pixels of low value (dark background) and the right side of the edge pixels of high value (bright face of the speaker). The simplicity of this approach makes it attractive, but the problems of the method are quickly visible. Apart from defining the measure for localisation every feature, the method also tries to describe relationships between the features (springs). Each spring joining \(i\)-th and \(j\)-th feature is given a cost function \(g_{ij}(x_i, x_j)\) dependent upon the 2D position of the two features it describes (denoted as \((x_i, x_j)\)). The best location of the 'spring model' on the actual scene is a 2D position at which the following 'fitness' measure is maximised:
\[ E = \sum_{i=1}^{2} x_i + \sum_{i=1}^{2} \left\{ \sum_{j \in N} g_j(x_i, x_j) \right\} \tag{3.7} \]

The inclusion symbol used in the second summation is due to the fact that not all of the elementary features are actually connected. The authors report that the system works on 35 from 40 images with most mistakes occurring in the localisation of mouth and nose.

One of the best known systems dedicated to facial feature extraction (as opposed to silhouette or face extraction) has been proposed in [yuille91]. The concept of deformable templates can be described as semantic based, since the shape of the templates of facial features (e.g. eyes, lips) is assumed before the processing starts (Figure 3.10).

![Eye template](image)

Similarly to the active contour (snake) approach the final shape of the template is dependent upon the 'forces' of the image, i.e. the deformable template is attracted to the areas of the image which minimise a certain energy function. In the proposed system, the template of the eye consists of four features, each of which would have its contribution to the fitting of the entire feature:

- A circle of radius \( r \), centred at a certain point \( X_e \) describing the boundary between the iris and the width of the eye. The 'cost function' for this feature is designed so that the interior of the circle is attracted to low pixel values and the circle itself is attracted to edges in the image intensity.
- Two parabolas describing the boundary of the white of the eye. The parabolas would have a common centre \( X_e \).
- Two points describing the centres of the white of the eye on the left and on the right of the iris. These two points would be attracted to centres of regions of high intensity.
- The region between the parabolas (describing the white of the eye itself) would have a cost function attracting it to the regions of high intensity on the image.
The location of the feature would be found by maximisation of total energy

\[ E_{TOT} = E_1 + E_2 + E_3 + E_4 \]  \hspace{1cm} (3.8)

where \( E_1 - E_4 \) denotes the energy due to placement of each of the sub-features of the deformable template of the eye. The fifth element denotes the energy due to the internal forces (forces keeping the four mentioned in one place, but allow rotation or scaling). In the description of the results it is mentioned that the algorithm is quite intensive computationally, but no conclusions are reached as to how robust the proposed method is.

A system partially based on deformable templates was presented in [craw92]. The system (called ‘FindFace’) aims to achieve automatic localisation of 40 feature points on the human face. Localisation of those points would enable extraction of both head outline and facial features contours (Figure 3.11).

![Figure 3.11: Extraction points: each vertex denotes an extraction point](image)

There are numerous proposals for sets of feature points (e.g. [kanade77, goldstein72]). In [craw92] the set described in [shepherd86] was used. The proposed two-step method uses separate modules for detection of individual facial features (i.e. each feature is found using a different mathematical tool). In step one, a rough estimation of the position of the speaker’s head is given; in step two the separate modules are invoked and allowed to process the image within very limited search areas. The system also introduces a concept of ‘experts’. Each expert can create its own solution to certain part of the problem.
The field of ‘expertise’ of the two experts can overlap, so that a certain redundancy can be introduced. The redundancy in finding a solution is welcome, since most of the experts fail in certain areas. The expert would normally be required to locate a certain feature and estimate its confidence in the newly estimated location. In order to find the outlines of the head a feature expert based on an approach similar to snakes was used. Here the head boundary would be detected by a ‘polygonal template’ [knoerr88]. The polygonal approach is more general since - unlike the snake - the polygonal template can be initialised anywhere on the image (a snake can only be initialised outside the analysed feature) and the background can be cluttered. Once the important facial features (the eyes, the lips) are detected, the remaining - less crucial - facial features are detected using corresponding experts. For example: the chin is located as the only strong edge crossing below the mouth, the cheeks are found when the smooth skin texture comes to an abrupt edge (assuming that background has a different texture). The hair is detected based on statistical evaluation, i.e. if there is sufficient difference in statistics describing two regions in the central face region (mean grey level, standard deviation, dynamic difference) they are separated as face and hair. Search for the position of the nose (which is not regarded as an important facial feature in all techniques) is based on template matching to the greyscale model of a single nostril. The system has been tested on a sequence of 64 images of a moving subject. The outline of the head was detected in 43 cases.

A lip location detector (in an image sequence) is proposed in [mak94]. While the motion model is similar to the one utilised in block DCT based moving image coding techniques, the lips themselves are located using morphological operators (all being straight derivatives of dilation and erosion) [maragos86, serra82] and cluster analysis. Similarly to the technique presented by [seferidis91] the resultant edge image (in this case of the lips) is created as a difference between morphologically pre-processed images (Figure 3.12).

A technique dedicated solely to tracking the movement of the eye (or more accurately - movement of the pupil) was proposed in [sung93]. A statistical algorithm based on maximum likelihood was employed here. The system was supposed to measure pupil size and position and rotation of the eye itself. It is claimed that the system was highly successful (over $10^7$ frames were tested). Since the system assumes that position of the eye is known a priori, it cannot exist on its own, but it might be considered as an end-stage step in an eye detection process.

65
All the methods presented above can be described as 'traditional'. Although some of them were proposed recently, they all share one common feature: they are not very reliable and their performance can be summarised as 'erratic'. Most of the reported methods are very complex and brittle. They are usually tuned to the test sequences or images they use. The value of these techniques for further research is therefore questionable.

### 3.3.2 Facial recognition techniques

As this thesis is not devoted to facial recognition, only the most promising techniques will be briefly explained here. In this section, systems based on ANNs (artificial neural networks) and PCA (principal component analysis) will be explained with the purpose of selecting the one most appropriate for supporting a knowledge based approach to coding videophone scenes at extremely low data-rates. At the present moment [chellappa95] ANNs and PCA seem to produce the best recognition results. A resurgence of interest in facial recognition has resulted from recent developments in computer technology because of the considerable increase of computing power of available equipment.
3.3.2.1 Artificial neural networks based systems

One of the basic tasks of ANNs is to be able to classify unknown data-sets, basing on training performed using a well-defined example set. Let us consider a simple example of ANN operation on small images (character recognition).

For example (Figure 3.13) a 'multi-layer perceptron' (MLP) [kulkarni94] ANN designed to recognise (classify) letters would have $M \times N$ input artificial neurons (where $M$ and $N$ are dimensions of the 'image' of the analysed character in pixels), 24 output artificial neurons and at least one hidden layer of artificial neurons. The network would be expected to produce a '1' on only one output artificial neuron. The remaining neurons would be expected to produce a '0'. Application of the image of the character 'A' to the input layer would be therefore expected to produce 'HIGH' on the first output neuron, 'B' would 'turn on' the second (and only the second) artificial neuron and so on (Figure 3.14).

![Figure 3.13: A multi-layer perceptron artificial neural network; each circle represents an artificial neuron, the arrows point in the direction of the signal](image)

![Figure 3.14: MLP applied to character recognition](image)
In the training phase the above can be achieved by adjusting of the weights of the specific artificial neurons during sequential application of all characters of the alphabet. After training the artificial neural network is expected to be able to generalise, i.e. to 'highlight' appropriate neurons even if the input characters are slightly corrupted.

Figure 3.15: Analysis of an image using artificial neural networks

The problems start when the size of the image increases. The system that performs the analysis could be subdivided into six stages (Figure 3.15): image acquisition, image pre-processing, feature extraction, associative storage, knowledge base and recognition. These stages can be classified into three groups: low-level processing, intermediate-level processing and high-level processing. In the acquisition stage, the image is digitised into a two-dimensional array of luminance (grey-scale) values. In other words both spatial co-ordinates and luminance are digitised. The next stage (pre-processing) is also referred to as the low-level processing. The following techniques are currently in use: enhancement, noise filtering, isolation of regions, geometric correction, restoration, reconstruction and segmentation. Image enhancement techniques can be further classified into spatial domain and frequency domain. The spatial domain restoration techniques operate directly on the grey-scale image while the frequency domain restoration modifies the Fourier transform of the image. Application of artificial neural networks is already possible at this stage (low-level processing: image restoration). From the point of view of image analysis the next stage (the intermediate stage) is the most important. Feature extraction is probably the best-known intermediate-level processing technique. The main task of feature extraction is to map the input image (two-dimensional array) onto the feature vector. The main reason behind feature
Chapter 3: Knowledge based coding and facial recognition

Knowledge extraction is to reduce the amount of data necessary for further processing. The main difficulty is finding the feature extraction technique that reduces the amount of data to be processed, but does not eliminate the data specific to the analysed image. The question of feature extraction will be discussed in more detail in the following paragraphs. Both 'traditional' and artificial neural networks application will be discussed. The last three stages: associative storage, knowledge base and the recognition are high-level processes.

Associative memory can be described as a content-addressable memory, i.e. it allows recovery of full information from data that is partially corrupted or that is known to represent only a certain part of the analysed data. Because the full information (or the expected information) is not known, a self-organising architecture is necessary here (e.g. a competitive learning artificial network). The task of recognition is classification. It assigns a label to an object based on the input information. Here a different computing architecture seems to be appropriate: the multi-layer perceptron network (MLP). However both supervised and unsupervised artificial neural networks are in use at this stage. The prior knowledge about the analysed data can be stored in a database. This knowledge database reduces the computational effort necessary for processing of similar data.

One of the most sophisticated neural network-based algorithms to detect frontal views of faces in grey-scale images was presented in [rowley95]. The system operates in two stages. First, a set of neural networks is applied to the analysed image. Then, an arbitrator is used to combine the outputs. The filter examines each location in the image, looking for locations that might contain a face. The arbitrator then merges outputs from individual filters and eliminates overlapping results. The first stage (the neural-network-based filter) receives an input 20x20 pixel region of the image, and generates an output ranging from 1 to -1, signifying the presence or absence of a face respectively. The filtering stage is applied at every pixel. However, before the image is applied to the neural network, pre-processing is performed in order to reduce the influence of lightning conditions and background luminance (Figure 3.16). First, the intensity values are equalised across the window. A function which varies linearly across the window is applied. The luminance that the linear function spans is defined by grey-scale levels of the pixels on the oval surrounding the analysed sub-image. The pixels outside the oval are discarded in order to reduce the influence of the background texture. The line function images are then subtracted from the original images. The resulting image is then histogram-equalised (Figure 3.16). The pre-processed image is then passed to a neural network. Each part of the ANN is designed to detect a different feature (mouths, pairs of eyes, individual eyes, nose, corners of the mouth, etc.) The neural network was trained.
using images from Harvard University. Multiple neural networks were applied. Each network was trained in a similar manner, but with random initial weights. Due to the biases of the individual networks, the networks rarely agree on a false detection of a face. An arbitration must be performed as the final stage of the analysis. The system was tested against a wide range of images. According to [rowley95], the system was able to detect 90.5% of the faces in a test set.

Figure 3.16: Pre-processing of 20x20 window (top to bottom): mask eliminating background pixels, original images, line function images, lightning corrected images, histogram equalised images.

Among tens of contributions looking at application of neural networks for facial recognition interesting examples can be found in [vincent95] (face localisation) and [golomb95] (sex recognition).

### 3.3.2.2 Principal components analysis based systems

The method of principal component analysis is primarily a data-analytic technique that obtains linear transformations of a group of correlated variables subject to certain optimal conditions being achieved. The most important of these conditions is that the transformed variables are uncorrelated. PCA considers two or more related random variables and as such is an important method in multivariate analysis. It also allows us to describe results (observations) with as few numbers as possible. The method of principal components was first proposed by Karl Pearson in 1901. However the general procedure as we know it today
was formulated by Harold Hotelling in 1933 [hotelling33]. It was Karhunen in 1947 [karhunen46] (followed by Loève in 1948 [loeve48]) who discovered a method of analogous transformation for transforming continuous data into a set of uncorrelated coefficients. The method of principal components is therefore often referred to as the 'Hotelling transform', or 'discrete Karhunen-Loève transform'. Sometimes the terms 'eigenvector transform' or 'principal component transform' are used as alternative names for the methods of principal components. Here, we deal with discrete signals only. Therefore the term 'Hotelling transform' or simply 'PCA' (principal components analysis) will be used.

Although PCA in image processing was first utilised for compression (an interesting example of which can be found in [welsh92]), it was facial recognition where it found its most successful application. Systems based on principal components analysis have been tested on data-bases of thousands, rather than hundreds of images [turk91, turk91-2, zhuje94, moghaddam95, pentland94]. The eigenface approach to face recognition has been inspired by research utilising PCA for efficient representation of pictures [sirovich87, kirby90]. Facial image analysis techniques are compared in [brunelli93]. Face recognition systems based on eigenfaces (eigenvectors of the covariance matrix of an ensemble of representative frontal face views) outperform algorithms using standard correlation methods. Eigenfaces offer more reliable and much faster processing of a facial data-base. Also, the PCA based techniques seem to be very robust against 'noise' (e.g. facial occlusions - Figure 3.17).

Figure 3.17: Test images (top), PCA-based recognition (bottom)
3.3.3 Summary

The review of recent papers dealing with the problem of localisation of the human face in images/video suggests, that there are two 'mainstreams' of research: a principal components approach and neural networks approach. Face finding algorithms based on the analysis of principal components have been tested on large image data-bases (in the range of thousands). On the other hand, methods based on neural networks have been tested on more complicated images, but the number of test images was much smaller (in the range of hundreds).

As well as providing good face detection performance, the eigenface approach offers the same mathematical tool for model-based compression of facial images and for facial recognition, which proves that PCA is a powerful, yet relatively simple tool. Neural networks seem to be less versatile here: they are simply trained as face detectors, not face classifiers.

It seems that PCA offers a more robust, simpler and computationally inexpensive tool to analyse facial images. There is no training involved and the only weakness of PCA seems to be its relative vulnerability to radical changes in luminance.

3.4 Conclusion

Knowledge based moving image coding techniques and facial recognition seem to share the same problems. Silhouette extraction, face localisation and facial features extraction are the problems still awaiting the ultimate solution, both in the facial recognition and model based coding areas. Obviously there are certain fundamental differences. In the model based coding system described in the following chapters, it is not essential to find a face which is the most similar to the one in the analysed scene.

Since the knowledge based approach offers potentially the highest compression ratio and since principal component analysis is the most promising facial recognition technique, a system for analysis of the scene in knowledge based coding using the principal components will be investigated.
4.1 Introduction

Automatic wire-frame fitting and automatic wire-frame tracking are the two most important and probably most difficult issues associated with semantic-based moving image coding. The focus of this chapter is automatic tracking of the motion in a 'head-and-shoulders' scene.

The successive frames of a *head-and-shoulders* video sequence are normally relatively similar to each other. All frames of such a sequence have one feature in common: they contain the face of the speaker. While classification of the image as a face does not present any problem for the human brain, it is extremely difficult for a machine to perform reliably the same task. However, an image of a face contains certain distinct features that can be identified automatically. Here, the efforts are concentrated on tracking these features: the left eye, the right eye, the nose and the lips, i.e. *important facial features*. Each facial feature should be tracked separately so that its 2D co-ordinates could be used to determine the current position of the speaker's head. This also allows implementation of the algorithm on a parallel machine. In this chapter a new method of tracking the position of important facial features for semantic wire-frame based moving image coding is proposed (Figure 4.1).

Figure 4.1: Tracking of important facial features
Reliable tracking of the facial features in head-and-shoulders scenes is of paramount importance for reconstruction of the speaker's motion in videophone systems if wire-frame based coding is used. The proposed method is based on eigenvalue decomposition of the sub-images extracted from subsequent frames of the video sequence. The motion of each facial feature (the left eye, the right eye, the nose and the lips) is tracked separately. No restrictions other than the presence of the speaker's face, were imposed on the actual contents of the scene. Although it was not the main goal of the research, the software-only tracking system works relatively fast. Due to the nature of the algorithm it should be relatively easy to implement it in hardware, or develop a parallel version of the software operating in real time.

The algorithm was tested on widely used head-and-shoulders video sequences containing moderate head pan, rotation and zoom. The results of these tests are presented in the final sub-sections of this chapter. The evaluation of the results is based on an error measure developed for estimation of the accuracy of tracking of the facial features in head-and-shoulders type of scenes.

This chapter focuses solely on scene tracking with the fitting problem covered in Chapter 5 and a unified approach to the problems of tracking and fitting being addressed in Chapter 6.

4.2 The tracking algorithm

A logical separation of the problem of wire-frame tracking and wire-frame fitting has proved to be a fruitful approach to the issues of semantic based moving image coding. Although this involved a temporary introduction of a 'human factor' in the analysis, the results achieved were sufficiently encouraging to make it worth following this path of analysis. Also, it allowed the entire approach to be more modular and flexible, i.e. applying separate techniques for wire-frame adaptation and for wire-frame fitting.

4.2.1 System preparation

For proper tracking of the motion of a 'head-and-shoulders' subject it is essential to recover the frame-by-frame position of its important facial features: the eyes, the lips, and the nose. The algorithm analyses the motion of each feature separately. Since, for the purpose of this analysis, a sequence of sufficiently similar images is necessary, how to obtain them will be described first.
Chapter 4: Tracking the subject in the scene

Figure 4.2: The left eye, the right eye, the nose and the lips sub-sequences

Figure 4.3: Extraction reference points for the sub-images of the eyes, the lips and the nose
Chapter 4: Tracking the subject in the scene

Four separate ‘sub-sequences’ of sub-images are created by manual extraction from the initial frames of the analysed head-and-shoulders sequence (Figure 4.2). Each sub-sequence contains sub-images of a different facial feature: the lips, the left and right eye, and the nose. These are referred to as the lips, the left eye, the right eye and the nose sub-sequence respectively. Each sub-sequence was created following certain guidelines that make the entire facial feature extraction process as objective and uniform as possible. The sub-images in the eyes, the lips and the nose sub-sequences were extracted with reference to certain reference points (Figure 4.3). In the case of the eyes the reference point is the centre of the iris, in case of the lips - the mid point between the left and the right corner of the lip, and finally for the nose - the mid point between the nostrils. The dimensions of all sub-images in a particular sub-sequence are the same. However, the dimensions of a sub-image in the sub-sequence containing one feature (e.g. the left eye) may be different from those in the sub-sequence containing another feature (e.g. lips). The subsequent analysis of each sequence is performed in the same way regardless of the size of the sub-image in the sequence.

![Figure 4.4: Scanning sub-images to obtain 1D vectors](image)

Once the extraction of the sub-sequences of the faces and the facial features is completed they are processed in the same way. Let us consider the sub-sequence of sub-images containing the left eye. Since the further analysis uses 1D vectors, each sub-image must be converted into such a form in a regular manner. The most straightforward method is to scan the image column-by-column, or line-by-line. Here all the sub-images are scanned line-by-line (for reasons of simplicity rather than any other). An image consisting of $R$ rows and $C$ columns would therefore produce a vector $\mathbf{x}$ consisting of $N = C \times R$ elements (Figure 4.4).
The sequence of scanned sub-images constitutes a matrix $X$ (referred to as the ‘input vector’ where each image - a vector itself - constitutes a single element of the input vector) (4.1).

$$X = [x_1 \ x_2 \ \ldots \ x_M]$$

$$Y = [y_1 \ y_2 \ \ldots \ y_M]$$

$$y_i = x_i - m_x$$

$$m_x = \frac{1}{M} \sum_{i=1}^{M} x_i$$ (4.2)

In the above equations $M$ denotes the number of sub-images in the sequence. The normalised input vector $Y$ is calculated by subtracting an expected value vector $m_x$ of the input vector $X$ from every single image of the sub-sequence (4.2).

The process of subtraction of the expected (mean) value of the sub-sequence can be interpreted as removal of the dc level. The graphical interpretation of the process is presented in Figure 4.5.

![Figure 4.5: Normalisation of the sub-sequence of the eyes](image)

4.2.2 Eigenvalue decomposition of a sequence of images

Principal component analysis is also referred to as the Hotelling [hotelling33] transform, or a discrete case of the Karhunen-Loève [karhunen46, loeve48] transform [gonzales77]. As will
shown here, the method of principal components is a reliable mathematical tool for tracking the motion of facial features in *head-and-shoulders* video sequences.

Let us consider a sub-sequence of $M$ sub-images extracted from the $M$ initial frames of a head-and-shoulders scene (4.1 - 4.2). In the first step of the method of principal components the eigenvectors of the covariance (dispersion) matrix $S$ of the sequence $X$ of $M$, $N$ dimensional input column vectors: $X = [x_1 \ x_2 \ ... \ x_M]$, $x_i = [x_{ij}]$, $j = 1..N$, $i = 1..M$, must be found. We can obtain the covariance matrix from the relationship (4.3).

$$S = YY^T$$

(4.3)

The columns of matrix $Y$ are the vectors $y_i = x_i - m_x$ and $Y^T$ is the transpose matrix of $Y$. Equation (4.3) can be expressed using equations (4.1 - 4.2), a symmetric non-singular matrix will be obtained (4.4),

$$S = \begin{bmatrix} s_1^2 & s_{12} & \cdots & s_{1N} \\ s_{12} & s_2^2 & \cdots & s_{2N} \\ \vdots & \vdots & \ddots & \vdots \\ s_{1N} & s_{2N} & \cdots & s_N^2 \end{bmatrix}$$

(4.4)

where $s_j^2$ is the variance of the $j$-th variable and $s_{jk}$, $j \neq k$, $(k = 1..N)$ is the covariance between the $j$-th and $k$-th variable. Since the images in the analysed sequence will be correlated, the elements of the matrix $S$ that are not on the leading diagonal will be non-zero. The objective of the method of principal components is to find the alternative co-ordinate system $Z$ for an input sequence $X$ in which all the elements other than the leading diagonal of the covariance matrix $S_Z$ (where index $Z$ denotes the new co-ordinate system) are zeros. According to matrix algebra such a matrix can be constructed if the eigenvectors of the covariance matrix are known (4.5).

$$S_Z = U^T S U$$

(4.5)

where $U = [u_1 \ u_2 \ ... \ u_N]$ and $u_j$, $j = 1..N$, is the $j$-th eigenvector of covariance matrix $S$. The eigenvectors can be calculated from the following relationship (4.6).

$$S u_j = \lambda_j u_j$$

(4.6)
where $l_j$ is the eigenvalue of the $j$-th eigenvector, $j = 1..N$.

As can be seen from equation (4.5), calculation of the eigenvectors involves operations on the covariance matrix $S$. Even if small images are used as an input sequence, the size of the covariance matrix can be too large to handle by common computing equipment (e.g. a sequence of images consisting of 100 columns and 100 rows would result in a $100^2 \times 100^2$ covariance matrix). However, if the number of images $M$ in the sequence $X$ is considerably smaller than the dimensions of the images themselves ($N = C \times R$), the above problem can be overcome. Let us consider a $M \times M$ matrix $C$ (4.7).

\[ C = Y^T Y \]  

(4.7)

The matrix $C$ has $M$ eigenvectors $v_i$, $i = 1..M$, and $M$ corresponding eigenvalues $m_i$ such that:

\[(Y^T Y) v_i = m_i v_i \]  

(4.8)

Pre-multiplying both sides of (4.8) by the matrix $Y$ and utilising relationship (4.3) we obtain:

\[ S (Yv_i) = m_i (Yv_i) \]  

(4.9)

Comparison of (4.6) and (4.9) suggests, that the eigenvectors of the covariance matrix $S$ can be expressed as linear combinations of the eigenvectors of matrix $C$. Since matrix $C$ is $M \times M$, the computational costs of finding the eigenvectors of the matrix $S$ are greatly reduced: in this algorithm $M < 20$. Thus the problem is reduced to calculations involving matrices smaller than $20 \times 20$. The vectors $v_i$ are also referred to as co-eigenvectors. The method of calculating eigenvectors via co-eigenvectors is referred to as singular value decomposition (SVD) [murakami82]. Using the nomenclature presented above, the Hotelling transformation can be expressed by (4.10).

\[ Z = U^T Y \]  

(4.10)

If we pre-multiply both sides of (4.10) by matrix $U$ and take into account the fact, that matrix $U$ is orthonormal (thus $U^T = U^{-1}$), the reverse Hotelling transformation will be obtained (4.11).
\[ Y = UZ \] (4.11)

The U matrix was derived using the input sequence X. If the analysed image was not originally a part of the input sequence, it can no longer expect to have an optimal principal component representation. However, principal component analysis can still be used for classification of unknown images which are relatively similar to the images from the input sequence. The difference between the unknown image and all the images used to generate the principal component space can be described by a specific distance measure. The ability to analyse unknown images using a fixed set of training images is utilised in face recognition algorithms.

Once the eigenvectors of the covariance matrix are calculated, they are stored as a pre-processed form of the feature sub-sequence they represent. Also, as will be shown subsequently, since it will not be necessary to store the sub-sequence itself, it is also possible to reduce storage requirements. However, the compression issues are not of primary interest here. Since the eigenvectors are sufficient to perform a forward and reverse Hotelling transform, they form a new processing space. This space is also referred to as 'principal component space' (PC space). The projection of the \( i \)-th image onto a PC space can be defined in the following way (4.12).

\[ z_i = U^T y_i \] (4.12)

In order to describe, for example, the PC space of the sub-sequence of the left eye, the term 'left eye PC space' will be used. Also, since the size of the eigenvectors is the same as the size of the images from the analysed sub-sequences, it is possible to visualise them (Figure 4.6). Visualised eigenvectors \((u_i)\) are generally referred to as eigen-images (or eigen-faces in case the analysed image contains a face). Each eigenvector has an eigenvalue (or latent roots) associated with it (4.6). The absolute value of an \( i \)-th latent root tells how relevant the \( i \)-th principal component is to the analysis of an image in the PC space. This also manifests itself in the shape of eigen-images. In Figure 4.6 the eigen-images of the sub-sequence of eyes are ordered (in a line-by-line manner) according to decreasing value of their latent roots. As can be seen, the images from the upper rows are similar to the differential versions of the images from the sub-sequences. The images in the lower rows describe changes that are common to an ever decreasing number of initial set images. This allows analysis of the parts of an image
'in order of importance'. This ability of PCA is of great importance for both compression and recognition purposes.

![Figure 4.6: Eigen-images of differential images presented in Figure 4.5](image)

### 4.2.3 Automatic tracking

In all the experiments described here the initial $M=16$ frames were used for the extraction of sub-sequences of important facial features. Thus automatic tracking starts in frame $M+1=17$. The initial position of the tracked facial feature in frame $M + 1$ (current frame) is assumed to be the same as in frame $M$ (previous frame). This view is subsequently verified in the following way (Figure 4.7).

![Figure 4.7: Processing of the current frame](image)
Sub-images within the search region are extracted from the current frame (e.g. a $15 \times 15$ search range would result in 225 sub-images) from the area twice the size of the sub-image from the analysed sub-sequence (in this case the sub-sequence of the left eye). In order to avoid confusion these images are referred to as the *extracted set*. The term *initial set* will be used to describe the sub-images extracted from the $M$ initial frames (matrix $X$) - Figure 4.8. The dimensions of the images from the extracted set are identical to those from the initial set. Since the extracted set images are similar to those from the initial set it can be assumed that they can be projected onto the principal component space created by input vector $X$. The further analysis of the extracted set is performed in $R$-dimensional ($R \leq M$) principal component space.

![Figure 4.8: Initial set and extracted set](image)

The goal is to find, among the images in the extracted set, the image that is most likely to contain the required feature. This image is referred to as a *best match* image. Its 2D co-ordinates will mark the position of the tracked facial feature in the current $(M + 1)$ frame. It is worth noting that the dimensionality of the principal component space can be reduced to less than $M$, as correct *recognition* of the most similar sub-image is required, rather than its reconstruction. Analysis of the variance of a particular principal component (i.e. the absolute value of the characteristic root associated with it) can lead to its elimination from further processing. This approach reduces further the execution time for the algorithm.

### 4.2.4 Best match measure and upgrading of initial set

The description of the algorithm given in the previous paragraph does not specify exactly what measure should be used for finding the best match image in the proceeding frame. Also,
Chapter 4: Tracking the subject in the scene

the necessity of updating the initial set should be thoroughly investigated. Both matters were investigated experimentally.

4.2.4.1 Distance measure

Two distance measures were taken into account. Since the algorithm is based on the analysis of the space of the principal components, the most straightforward method of establishing the difference between two images is calculation of the Euclidean distance separating them in the principal component space created by matrix \( \mathbf{X} \) (referred to as 'distance measure A' - (4.13)).

\[
e_{ij} = \| \mathbf{a}_i - \mathbf{b}_j \| \quad (4.13)
\]

where \( \mathbf{a}_i \) is the projection of the \( i \)-th image vector from the initial set onto \( K \)-dimensional principal component space according to relationship (4.10), and \( \mathbf{b}_j \) is the projection of the \( j \)-th image from the extracted set onto the same space. The distance \( e_{ij} \) is calculated for every possible pair of images from the initial set and the extracted set. The \( j \)-th image from the extracted set for which the \( e_{ij} \) distance is minimal is the best match image. Its 2D co-ordinates mark the position of the particular facial feature in the next frame.

A different method of derivation of the best match image has also been tested. The best result found using relationship (4.13) points to the image which is the most similar to one particular image from the initial set. However, it seems that finding the image that is most similar to all the images from the initial set might provide a better measure of distance. A suitable distance measure can be obtained using the following equation: (referred to as 'distance measure B').

\[
d_i = \| y_i - \sum_{j=1}^{K} z_j u_j \| \quad (4.14)
\]

\[
z_j = u_j^T (x_j - m) \quad (4.15)
\]

where \( y_i \) denotes the normalised \( i \)-th image from the extracted set, \( u_j \) - the \( j \)-th eigenvector of covariance matrix \( \mathbf{S} \) and \( z_j \) - the \( j \)-th principal component. The more similar the unknown image is to the initial set, the smaller will be the distance \( d_i \). Large differences between the
input and output images will result in a large value of \( d_i \). Similarly to the previous case, the \( i \)-th image from the extracted set for which the \( d_i \) distance is minimal is the best match image and its 2D co-ordinates mark the most likely position of the particular facial feature in the next frame.

4.2.4.2 Upgrading of the initial set

The initial set (the sub-images containing facial features extracted from the initial frames) can be updated throughout the sequence, i.e. certain sub-images that were added to the set before automatic tracking started, can be replaced dynamically by new images.

![Figure 4.9: Dynamic upgrade of the initial set](image)

Should this happen, the principal components space would have to be re-calculated in order to accommodate the replacement sub-image. Although this involves additional computational load, inclusion of a best match into the initial set on a frame-by-frame basis might prove advantageous. The following technique for initial set upgrading was tested.

Once the best mach sub-image is found in the current frame (either using (4.13) or (4.14)) it is most likely to contain a desired facial feature. Should this be the case, its contents would be more similar to the same facial feature in the following frame, than, e.g. the first sub-image in the manually extracted initial set. For this reason the best match sub-image in the current frame would replace the sub-image extracted from the most ‘remote’ frame of the sequence (Figure 4.9). In this way after \( M' = 2 \times M \) all the sub-images from the first initial set would be replaced by best matches established on the way.
4.2.4.3 Testing method

In order to find out which of the two distance measures is more appropriate for the purposes of tracking of the geometrical centres of important facial features, tests involving tracking separately the left eye, the right eye, the nose and the lips of the subject in Miss America sequence (CIF, 352x240 pixels, 150 frames) using distance measure A and distance measure B (test B) were carried out (Figure 4.10). For each distance measure static and dynamic (updated on frame-by-frame basis) initial sets were tested. In total, four tests were carried out on all important facial features. In order to test the quality of tracking, movies with white crosses centred on the tracked facial features (tracking crosses) were created (e.g. Figure 4.11).

![Figure 4.10: Testing error measures on sub-image sequences extracted from Miss America](image)

Figure 4.10: Testing error measures on sub-image sequences extracted from Miss America

![Figure 4.11: Dynamically updated initial set, distance A, frame 45 and 48: eye tracking](image)

Figure 4.11: Dynamically updated initial set, distance A, frame 45 and 48: eye tracking

In the first instance, the algorithms utilising the dynamically updated initial set were tested. Disappointingly, both application of distance A and distance B did not deliver the desired performance. Although the method based on distance B proved more robust, in both cases tracking was not maintained throughout the entire sequence for some of the features. In the
case of distance A, the tracking was successfully maintained for the left eye and the nose. The track of the right eye was lost in frame 48 (Figure 4.11), while tracking of the lips was rather erratic (the tracking cross did not follow the middle of the lips, Figure 4.12).

When measure \( B \) was used, the tracking was maintained for the left eye, the right eye, and the nose (Figure 4.13). The track of the lips was lost in frame 110 (Figure 4.14).
Although the similarity between subsequent frames is indisputable and it seems reasonable to upgrade the initial set by addition of most ‘recent’ sub-images, the algorithm does not guarantee that the best match will be found for every single frame. Incorrect classification is possible and clearly took place in both cases. If incorrect recognition takes place, the erroneously chosen best match is immediately used to re-calculate the principal components space. Since the sub-image from the following frame will most likely be very similar to the sub-image extracted from the previous frame, the distance (calculated using (4.13)) will be very small. This is because the distance measure $A$ calculates the distance between two distinct sub-images: one from the extracted set, and the other from the initial set. It is therefore very likely that the two sub-images giving the smallest value in (4.13) will have originated in adjacent frames. This makes the tracking algorithm unstable and prone to noise. Tracking would ‘drift away’ in the direction of the first erroneous best match recognition. Although still unsuccessful, application of a dynamic initial set and measure $B$ proved more robust, since the shape of (4.14) ensures, that the feature is compared to the entire initial set. However, the influence of the erroneously added sub-image apparently re-shapes the entire PCA space reducing correct classifications.

On the other hand, tracking of the sequence using a static initial set proved very robust. In experiments involving both measure $A$ and measure $B$ the tracking was maintained throughout the entire sequence for all facial features. The centres of the tracking crosses were constantly kept within the corona of the eye (when visible), between the nostrils, and in the centre of the lips. Tracking was maintained even when the speaker closed her eyes (quite frequently in the Miss America sequence) or opened and closed her lips.

However, direct comparisons between the two methods proved that the tracking using a static set and the distance measure $B$ was more regular and smooth. On several occasions, even in
early frames, the difference in the quality of tracking was quite evident (Figure 4.15 - Figure 4.18). When distance measure $B$ was used, the motion of tracking crosses reflected the motion of facial features with great fidelity, also the response to head pan was, unlike in tests involving measure $A$, immediate.
Chapter 4: Tracking the subject in the scene

Figure 4.19: Error measured as the Euclidean distance in image space

In order to assess the accuracy of the tracking method quantitatively, the 2D locations of the important facial features were extracted manually from all 150 frames of the Miss America sequence. The error was measured as the absolute of the Euclidean distance between the 2D co-ordinates of the features tracked automatically (using measures A and B) and manually (Figure 4.19) on a frame-by-frame basis yielding an error profile for the entire sequence. Tracking error profiles are presented (Figure 4.20 - Figure 4.23). In all cases, tracking the motion of facial features using measure B is far more stable. Tracking the motion of the lips using measure A seems to be particularly unstable and prone to erroneous recognition.

Figure 4.20: Measure A (left) and measure B (right) left eye tracking error profiles

Figure 4.21: Measure A (left) and measure B (right) right eye tracking error profiles
However the change in shape of the lips does not seem to have any major influence on tracking using measure B. Since the application of the dynamic initial set failed, only the statistical results of tests using distance measure A and distance measure B with static initial set are presented here (Table 4.1). As can be seen, the numerical error evaluation using Euclidean distance confirms the subjective result: measure B is more stable and delivers better overall performance.

<table>
<thead>
<tr>
<th>Facial feature</th>
<th>Measure A</th>
<th>Measure B</th>
</tr>
</thead>
<tbody>
<tr>
<td>Left eye</td>
<td>0.71</td>
<td>0.61</td>
</tr>
<tr>
<td>Right eye</td>
<td>1.32</td>
<td>1.23</td>
</tr>
<tr>
<td>Nose</td>
<td>1.23</td>
<td>0.83</td>
</tr>
<tr>
<td>Lips</td>
<td>1.83</td>
<td>1.37</td>
</tr>
</tbody>
</table>

Table 4.1: Mean tracking error and standard deviation: Miss America (150 frames)
Chapter 4: Tracking the subject in the scene

The mean error in tests involving measure \( B \) is less than one pixel for all the tracked facial features. Measure \( A \) describes the distance in principal component space between two individual images (one from the initial set and one from the extracted set). On the other hand, measure \( B \) describes the difference between the normalised image from the extracted set and its reconstruction. As the reconstruction is carried out using eigenvectors of the principal components space created using all images from the initial set, this latter measurement is likely to give a better indication of the similarity to all the images from the initial set (Figure 4.24).

Figure 4.24: Measure \( B \) describes the distance from a set of images rather than an individual image

Since the initial set is not updated, an accidental erroneous classification is unlikely to cause more than a glitch that is rectified after a couple of frames. This effect is visible in the case of tracking the motion of the right eye and the lips using measure \( B \) (Figure 4.21, Figure 4.23). Because of its superior performance measure \( B \) was employed in all subsequent tests.

4.2.4.4 Reducing the dimensionality of the principal component space

Tracking a single facial feature using the method proposed in previous paragraph was carried out relatively fast using software tools only. The approximate tracking speed was two frames per second using a machine with a 120MHz Pentium processor. Since the project involved a multitude of various tracking and fitting techniques, optimisation of the code was never of high priority. However, in the case of PCA the issue of processing speed is associated with an interesting problem concerning reducing the dimensionality of the principal components space. Reduction in the dimensions of the transformed space is theoretically possible due to the fact that the initial principal components (i.e. principal components related to roots of higher absolute values) are likely to account for most of the variations in the scene. In many
Chapter 4: Tracking the subject in the scene

In some cases it is therefore possible simply to discard the remaining principal components and thus reduce the dimensionality of the principal component space.

In the following figures (Figure 4.26 - Figure 4.31) the sub-images, differential images and eigenvector images of some of the sequences are presented. While the sub-sequence of facial features in the composite image is presented in natural order (i.e. first sub-image is extracted from the first frame of the sequence, second sub-image from the second frame etc.), the eigenvector images are arranged in order of decreasing 'importance', i.e. in order of decreasing associated latent, or characteristic root (Figure 4.25).

Figure 4.25: Eigenvector images arranged in order of decreasing 'importance'

Figure 4.26: Claire initial sets for the eyes, the nose and the lips

Figure 4.27: Claire difference images for the eyes, the nose and the lips

Figure 4.28: Claire eigenvector images for the eyes, the nose and the lips
As can be seen, whereas the top rows of the composite eigenvector images present certain meaningful combinations of greyscale pixels, the bottom rows tend to contain rather uncorrelated data (apart from the lips in Figure 4.28 where the upper and lower lip is still distinguishable in the bottom row). This might lead to the conclusion that it is possible to drop certain principal components from the analysis entirely. In this particular research the four least meaningful principal components were dropped from the analysis. Although tests involving truncating larger numbers of components were carried out, these will not be presented here since speed and compression are not the main concerns of this thesis.

4.2.5 Results of tests on head-and-shoulders video scenes

A number of widely used head-and-shoulders video sequences were used for further testing. These are listed in Table 4.2, and are readily available from well known WWW sites. They reflect a wide variety of possible head-and-shoulders videophone situations containing moderate speaker’s head pan, rotation and zoom.
Chapter 4: Tracking the subject in the scene

<table>
<thead>
<tr>
<th>Sequence</th>
<th>Horizontal size [pixels]</th>
<th>Vertical size [pixels]</th>
<th>Length [frames]</th>
</tr>
</thead>
<tbody>
<tr>
<td>Miss America</td>
<td>352</td>
<td>240</td>
<td>150</td>
</tr>
<tr>
<td>Claire</td>
<td>360</td>
<td>288</td>
<td>168</td>
</tr>
<tr>
<td>Car Phone</td>
<td>176</td>
<td>144</td>
<td>400</td>
</tr>
<tr>
<td>Grandma</td>
<td>176</td>
<td>144</td>
<td>768</td>
</tr>
<tr>
<td>Salesman</td>
<td>360</td>
<td>288</td>
<td>400</td>
</tr>
<tr>
<td>Trevor</td>
<td>256</td>
<td>256</td>
<td>100</td>
</tr>
</tbody>
</table>

Table 4.2: Sequences tested

CIF-sized sequences (*Miss America, Claire, Salesman, Trevor*) as well as QCIF sequences (*Car Phone, Grandma*) are included. The subjects of two video sequences (*Trevor, Grandma*) wear glasses and the *Car Phone* sequence was shot in a moving car.

The algorithm produced consistent results for all tested sequences. In all tests tracking of all important facial features was maintained throughout the entire sequence. In order to obtain a
numerical measure of the accuracy of the algorithm, a method similar to the one applied for distance measure testing was used. However, because of the length and the number of test sequences, the 2D positions of the important facial features were extracted manually from every fifth frame of each sequence only. As can be seen from Figure 4.32, this sampling can still be regarded as representative (Table 4.3).

For each facial feature and each test sequence an error profile was constructed in order to display the differences between manual and automatic target tracking (Figure 4.33 - Figure 4.44). The quality of tracking of a particular facial feature can be described by the value of the mean and the standard deviation of its error profile. These are also listed here (Table 4.4 and Table 4.6). As can be seen, the mean error for all the facial features in almost all the sequences was less than 1 pixel. The left and the right eye seem to be the most reliably tracked facial features, possibly due to the combination of very light (white of the eye) and very dark (corona of the eye) pixels. However, track was also maintained during periods when the speakers closed their eyes. There were no problems with tracking eyes partially occluded by glasses. Especially in the case of the QCIF sequences (Car Phone, Grandma) the reference manual fitting can also introduce additional error since it is sometimes difficult to judge the exact position of the middle of the particular facial feature.

<table>
<thead>
<tr>
<th>Sequence</th>
<th>Resulting sample [frames]</th>
</tr>
</thead>
<tbody>
<tr>
<td>Miss America</td>
<td>30</td>
</tr>
<tr>
<td>Claire</td>
<td>34</td>
</tr>
<tr>
<td>Car Phone</td>
<td>80</td>
</tr>
<tr>
<td>Grandma</td>
<td>154</td>
</tr>
<tr>
<td>Salesman</td>
<td>80</td>
</tr>
<tr>
<td>Trevor</td>
<td>20</td>
</tr>
</tbody>
</table>

Table 4.3: Manual extraction sample

However, the overall results clearly demonstrate that the algorithm is able to maintain tracking of the important facial features for a variety of head-and-shoulders sequences. Stills which illustrate the range of movement of the facial features (Figure 4.45 - Figure 4.68) in the test sequences follow the error profiles.
Figure 4.33: Miss America: The left eye (left) and the right eye (right) tracking error profiles

Figure 4.34: Miss America: The nose (left) and the lips (right) tracking error profiles

Figure 4.35: Claire: The left eye (left) and the right eye (right) tracking error profiles

Figure 4.36: Claire: The nose (left) and the lips (right) tracking error profiles
Chapter 4: Tracking the subject in the scene

Figure 4.37: Car phone: The left eye (left) and the right eye (right) tracking error profiles

Figure 4.38: Car Phone: The nose (left) and the lips (right) tracking error profiles

Figure 4.39: Grandma: The left eye (left) and the right eye (right) tracking error profiles

Figure 4.40: Grandma: The nose (left) and the lips (right) tracking error profiles
Chapter 4: Tracking the subject in the scene

Figure 4.41: *Salesman*: The left eye (left) and the right eye (right) tracking error profiles.

Figure 4.42: *Salesman*: The nose (left) and the lips (right) tracking error profiles.

Figure 4.43: *Trevor*: The left eye (left) and the right eye (right) tracking error profiles.

Figure 4.44: *Trevor*: The nose (left) and the lips (right) tracking error profiles.
## Chapter 4: Tracking the subject in the scene

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Left eye</td>
<td>0.62</td>
<td>0.68</td>
<td>0.41</td>
<td>0.53</td>
</tr>
<tr>
<td>Right eye</td>
<td>0.84</td>
<td>0.69</td>
<td>0.64</td>
<td>0.74</td>
</tr>
<tr>
<td>Nose</td>
<td>0.57</td>
<td>0.60</td>
<td>0.78</td>
<td>0.55</td>
</tr>
<tr>
<td>Lips</td>
<td>0.47</td>
<td>0.58</td>
<td>1.02</td>
<td>0.62</td>
</tr>
</tbody>
</table>

Table 4.4: Tracking error results for Miss America and Claire

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Left eye</td>
<td>0.88</td>
<td>0.72</td>
<td>0.86</td>
<td>0.60</td>
</tr>
<tr>
<td>Right eye</td>
<td>0.89</td>
<td>0.78</td>
<td>0.83</td>
<td>0.65</td>
</tr>
<tr>
<td>Nose</td>
<td>1.06</td>
<td>0.87</td>
<td>0.37</td>
<td>0.53</td>
</tr>
<tr>
<td>Lips</td>
<td>1.66</td>
<td>1.04</td>
<td>0.73</td>
<td>0.67</td>
</tr>
</tbody>
</table>

Table 4.5: Tracking error results for Car Phone and Grandma

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Left eye</td>
<td>0.86</td>
<td>0.78</td>
<td>0.84</td>
<td>0.75</td>
</tr>
<tr>
<td>Right eye</td>
<td>0.83</td>
<td>0.96</td>
<td>0.84</td>
<td>0.89</td>
</tr>
<tr>
<td>Nose</td>
<td>0.85</td>
<td>0.69</td>
<td>0.85</td>
<td>0.58</td>
</tr>
<tr>
<td>Lips</td>
<td>0.94</td>
<td>1.31</td>
<td>0.70</td>
<td>0.70</td>
</tr>
</tbody>
</table>

Table 4.6: Tracking error results for Salesman and Trevor
Figure 4.45: Tracking Miss America; frames: 0, 20

Figure 4.46: Tracking Miss America; frames: 85, 98

Figure 4.47: Tracking Miss America; frames: 110, 120

Figure 4.48: Tracking Miss America; frames: 124, 140

Chapter 4: Tracking the subject in the scene
Chapter 4: Tracking the subject in the scene

Figure 4.49: Tracking Claire; frames: 0, 75

Figure 4.50: Tracking Claire; frames: 95, 105

Figure 4.51: Tracking Claire; frames: 140, 145
Chapter 4: Tracking the subject in the scene

Figure 4.52: Tracking Claire; frames: 155, 167

Figure 4.53: Tracking Grandma; frames: 30, 40, 70

Figure 4.54: Tracking Grandma; frames: 120, 180, 280

Figure 4.55: Tracking Grandma; frames: 340, 410, 440

Figure 4.56: Tracking Grandma; frames: 500, 600, 700
Chapter 4: Tracking the subject in the scene

Figure 4.57: Tracking Car Phone; frames: 30, 55, 75

Figure 4.58: Tracking Car Phone; frames: 80, 105, 170

Figure 4.59: Tracking Car Phone; frames: 200, 290, 320

Figure 4.60: Tracking Car Phone; frames: 333, 377, 399

Figure 4.61: Tracking Salesman; frames: 0, 50
Figure 4.62: Tracking Salesman; frames: 80, 120

Figure 4.63: Tracking Salesman; frames: 140, 210

Figure 4.64: Tracking Salesman; frames: 340, 351
Chapter 4: Tracking the subject in the scene

Figure 4.65: Tracking Trevor; frames: 0, 20

Figure 4.66: Tracking Trevor; frames: 30, 40

Figure 4.67: Tracking Trevor; frames: 50, 70

Figure 4.68: Tracking Trevor; frames: 80, 99
Since the tested sequences included both CIF and QCIF-sized examples, the error distance was normalised by dividing its absolute value by the distance between the centres of the left and right eye. Subsequently the accuracy of tracking was judged based on the value of the normalised error distance (NED). If NED is less than 0.05, the tracking is described as very good on a particular frame. For normalised distances between 0.05 and 0.10 the tracking is described as good. Outside these boundaries, the track can be judged as satisfactory. The percentage of frames with tracking classified as very good, good, and satisfactory for each tested sequence and facial feature, is presented in Table 4.7 and Table 4.8.

<table>
<thead>
<tr>
<th>Left eye track [%]</th>
<th>Right eye track [%]</th>
</tr>
</thead>
<tbody>
<tr>
<td>V. good</td>
<td>Good</td>
</tr>
<tr>
<td>Miss America</td>
<td>96.3</td>
</tr>
<tr>
<td>Claire</td>
<td>93.5</td>
</tr>
<tr>
<td>Car Phone</td>
<td>67.5</td>
</tr>
<tr>
<td>Grandma</td>
<td>94.1</td>
</tr>
<tr>
<td>Salesman</td>
<td>51.9</td>
</tr>
<tr>
<td>Trevor</td>
<td>82.4</td>
</tr>
</tbody>
</table>

Table 4.7: The left and the right eye tracking quality

<table>
<thead>
<tr>
<th>Nose track [%]</th>
<th>Lips track [%]</th>
</tr>
</thead>
<tbody>
<tr>
<td>V. good</td>
<td>Good</td>
</tr>
<tr>
<td>Miss America</td>
<td>100.0</td>
</tr>
<tr>
<td>Claire</td>
<td>83.9</td>
</tr>
<tr>
<td>Car Phone</td>
<td>64.9</td>
</tr>
<tr>
<td>Grandma</td>
<td>98.7</td>
</tr>
<tr>
<td>Salesman</td>
<td>42.9</td>
</tr>
<tr>
<td>Trevor</td>
<td>82.4</td>
</tr>
</tbody>
</table>

Table 4.8: The nose and the lips tracking quality
4.3 Local motion tracking

The effectiveness of the algorithm described in this chapter for tracking the shape of individual facial features (local motion) is examined in this section. In the case of the eyes, the eye-brows and the lips this involves tracking of at least four vertices. The algorithm, although unchanged was initialised with a different kind of images. This time the initial set images were centred on the points of the image that corresponded to the positions of the wire-frame vertices of a particular facial feature (Figure 4.69, Figure 4.70 - Figure 4.75). Even if the shape of the facial feature changes radically, the contents of the image from the initial set will change only slightly. This assures continuity of tracking. The other advantage of a decrease in size of the images from the initial set is reduction of the computational load required to track the particular vertex.

Figure 4.69: Sub-images centred on the vertices describing facial features

Figure 4.70: Miss America initial sets for the left and right eye corners

Using this approach it was possible to track the closing and opening of lips and eyes. Results demonstrating tracking results on the example of Miss America (this time both with crosses tracking the centres of the important facial features and crosses tracing the local motion of the important facial features) are presented (Figure 4.76 - Figure 4.78).
Chapter 4: Tracking the subject in the scene

Figure 4.71: *Miss America* differential images for the left and right eye corners

Figure 4.72: *Miss America* eigenvector images for the left and right eye corners

Figure 4.73: *Miss America* initial sets for the lips

Figure 4.74: *Miss America* differential images for the lips

Figure 4.75: *Miss America* eigenvector images for the lips
4.4 Summary

In this chapter a technique for automatic tracking of facial features in head-and-shoulders sequences has been presented and tested. The results of the tests suggest that the method of principal components is a very reliable tool for tracking of very fine changes - including the local motion of the facial features (eyes open-close, lips open-close). The principal
component analysis is performed on images extracted from the analysed sequence, which seems to improve the reliability of tracking considerably.

Several variations of the algorithm were considered. Versions with static and dynamic initial sets were tested. Also, two distance measures were compared (measure $A$ describing distance between two individual images, and measure $B$ describing the distance between an image and a group of images). Since the algorithm is based on the analysis of principal components, reduction of the dimensionality of the principal component space was taken into account. Finally, tracking of the local motion (motion of the important facial features) was tested.

The tests proved that the version utilising dynamically updated initial set does not provide expected results. The track of certain facial features was lost on numerous occasions when distance measure $A$ was used. Although application of the distance measure $B$ brought improvements to the overall robustness of the tracking, it was not possible to recover the motion of all the facial features with sufficient reliability.

Substantial improvements in the tracking fidelity were possible after application of a static initial set. Application of both measures $A$ and $B$ allowed tracking of all facial features throughout the entire sequence. In order to assess the tracking accuracy in both methods, the error profiles depicting Euclidean distance between the 2D co-ordinates of the features tracked automatically and manually were created. The analysis of the mean error and the standard deviation proved that measure $B$ allows more accurate tracking. The robustness of the algorithm utilising measure $B$ and static initial set was confirmed as a result of tests on a range of widely used head-and-shoulder sequences. Tracking was maintained for all the important facial features in all the tested sequences.

The presented algorithm assumes manual extraction of features from the initial frames of the sequence. In the following chapter a method of automatic fitting is presented. It leads to an algorithm performing both fitting and tracking without on-line human intervention.
5.1 Introduction

In the previous chapter a solution to the problem of feature tracking was proposed. However, it was assumed that the position of the scene model had been known in the initial frames of the sequence. The subject of this chapter is automatic fitting of a generic wire-frame model \textit{(Candide [forchheimer89])}, i.e. wire-frame fitting without 'human intervention'. Accurate model fitting is the task of the encoder and is of particular importance to the fidelity of reconstruction of the scene at the decoder side. Although manual wire-frame fitting might be helpful for the analysis of tracking problems, it is not acceptable as a part of an operational videophone system. Model fitting must be performed automatically.

The sub-sections of this chapter present several initial heuristic approaches to the problems of automatic wire-frame fitting. This is followed by descriptions of several algorithms capable of \textit{Candide} wire-frame fitting. Each technique has been tested on a range of images. The results of these tests are included in this chapter. The most effective fitting algorithm is chosen and used as a base for automatic wire-frame tracking (described in Chapter 4). The accuracy of the model fitting algorithm should not be judged in terms of PSNR. Indeed, the results are presented in a comparative manner, where the reference (ideal) fit is performed by a human operator, rather than in terms of PSNR. Immediate comparison of manual and automatic fitting of the same wire-frame to the same subject offers a more meaningful and more realistic interpretation of the results. Good fitting results can be obtained even when PSNR values are high. Also, since the reference fitting is performed manually, the whole idea of calculating the error using PSNR is questionable.

Wire-frame fitting is a difficult task. It must be performed very accurately. This is of great importance in areas occupied by the facial features (the eyes, the lips, the nose). Whereas the fit for certain facial features does not have to be ideal (e.g. nose), the fit in the area of the
subject's face occupied by the lips must be very accurate. The issue of accurate fitting in the lips' area will receive particular attention in this chapter.

Although some of the less successful techniques described in this chapter (subjects of sections 5.2 and 5.3.2) were developed before the PCA-based method for tracking (described in Chapter 4) was verified to give very good results, it was decided that the chapter describing automatic fitting algorithms should follow the one describing automatic tracking algorithms. This is because the PCA-based fitting (subject of section 5.3.3 of this chapter) method was developed as a result of very good tracking fidelity obtained during testing of the algorithm from Chapter 4. Thus, the content of the section 5.3.3 is the result of experiences obtained both during developing of the tracking algorithm utilising PCA (previous chapter) and the fitting algorithm utilising a data-base of facial images (current chapter).

5.2 Initial attempts: feature extraction

As was mentioned previously, fitting of the wire-frame is of particular importance in the area occupied by facial features, especially in the lips' area. Therefore early experiments were concerned with extraction of the lips' area. Two generic techniques were developed and tested. This sub-section contains a description and the results from the application of both of them.

5.2.1 Lips centre extraction from motion

The motion in the lips' area is described by motion of the upper and lower lip. It is worth noticing that the upper lip performs motion in the opposite direction to the lower lip.

Figure 5.1: Subsampling the area occupied by the lips
This should produce a very clear motion field allowing separation of lower and upper lip and thus accurate localisation of the lips' area. Of course, the above is true only if the subject in the scene is talking.

Since this reasoning is based on a motion field, it was necessary to examine a sequence of at least two frames. In the preliminary experiments the *Miss America* sequence was used. Motion was analysed in the lips' 'search region' (Figure 5.1) The motion analysis was carried out as follows. The image was subsampled so that a 'pixel' (here referred to as 'macropixel') in the new image has four times the area of the pixel from the original one. The macropixel $L(x,y)$ (Figure 5.2) was calculated as a weighted sum of four pixels from the original image $I(x,y)$ (5.1).

![Figure 5.2: Macropixel](#)

$$L(x,y) = \frac{1}{2}I(x,y) + \frac{1}{5}I(x+1,y) + \frac{1}{5}I(x,y+1) + \frac{1}{10}I(x+1,y+1) \quad (5.1)$$

Subsequently, for each macropixel in the current frame the most similar macropixel from the previous frame was searched for. The area of the search region for each macropixel was fixed to two macropixels each way (Figure 5.3). Such a relatively small search region is dictated by the fact that lip motion can be regarded as 'smooth', i.e. it is very unlikely the change in lip position would be greater than a couple of pixels. The displacement between the analysed macropixel and the reference macropixel defines the 'motion pixel'. A motion pixel has four values attached to it: spatial co-ordinates, displacement co-ordinates and a greyscale level. The spatial co-ordinates are the same as the macropixel motion of which it describes. The displacement co-ordinates describe the difference in spatial positions of the analysed macropixel and its best match in the previous frame. Only two values of greyscale are allowed: 0 and 255.
If the $y$ motion co-ordinate is positive, the motion pixel is assigned grey level 255, otherwise it is assigned grey level 0. Since the resolution of the motion field is equal to that of subsampled version of the search area, the complete motion field can be visualised by motion pixels (Figure 5.4 - Figure 5.9).

Figure 5.3: Macropixel search area

Figure 5.4: Frame 16 of Miss America sequence (left) with superimposed motion field (right)

Figure 5.5: Frame 17 of Miss America sequence (left) with superimposed motion field (right)
Chapter 5: Fitting of a generic wire-frame model to the scene

Figure 5.6: Frame 18 of Miss America sequence (left) with superimposed motion field (right)

Figure 5.7: Frame 19 of Miss America sequence (left) with superimposed motion field (right)

Figure 5.8: Frame 20 of Miss America sequence (left) with superimposed motion field (right)

Figure 5.9: Frame 21 of Miss America sequence (left) with superimposed motion field (right)
A distinct edge between the upper and lower lip parts of the motion fields can be clearly seen. This system can be viewed as edge detector based on analysis of the motion. However, it must be said, that the above algorithm assumes that the motion of the subject is translational only. Also, global motion of the speaker must be established in advance. Still, the accuracy of centre of the lips recovery is very satisfying and the above algorithm might be used for fine estimation of lips' centre once its position in the scene was roughly established.

### 5.2.2 Segmentation and mathematical morphology

Effective segmentation of a still image from a head-and-shoulders type sequence could lead to extraction of the areas occupied by important facial features. The basic problem of feature extraction could then be reduced to the seemingly less difficult problem of scene segmentation. This led to further investigation by writing generic segmentation software capable of extraction of important facial features in a head-and-shoulders scene.

![Image segmentation approaches](image.png)

The segmentation algorithm should normally be proceeded by an image 'simplification' front-end (Figure 5.10). The most commonly used image simplification is low-pass filtering. This approach however causes loss of edges, normally associated with higher spatial frequencies.

Mathematical morphology is a relatively new branch of analysis as far as images are concerned [giardina88]. It was applied in late 1960 for black-and-white image processing. Later on, the mathematical model was extended to include greyscale imagery. Image filters based on mathematical morphology are claimed to be able to remove superfluous data from the image while maintaining edges. Morphology was used as a pre-processing step for object-based coding segmentation in [cortez95] and [brigger95]. In object-based coding techniques, segmentation is used for division of the image into objects, which do not necessarily have a
physical interpretation. In the ideal case, each segment should have a distinct physical meaning, i.e. a segment representing the eye, a segment representing the lips, etc. However, the results presented in [cortez95] suggest that it is possible to both segment an image effectively and obtain physically meaningful results. Obviously from the point of view of a semantic-based approach the latter result is very attractive, since application of segmentation proceeded by mathematical morphology based filtering might allow extraction of important facial features: the eyes, the lips and the nose.

Once pre-processing of the image is completed, the segmentation proper is applied. There are basically two methods of image segmentation (Figure 5.10). In the region growing technique, the algorithm starts with the location of 'seeds'. Each seed is treated as potential region. Adjacent pixels are subsequently assigned to a particular seed if the homogeneity criterion is satisfied. If all the pixels of the image are assumed to be seeds, the region growing technique may prove very expensive computationally. Region splitting techniques seem to be a more attractive alternative [horowitz76]. First the image is treated as a single region. Then four sub-regions are considered. If they do not satisfy the homogeneity criterion, they are split. From that point the four regions are analysed separately in the same way that the entire image was analysed. The splitting stops if the homogeneity criterion is satisfied for the four sub-regions considered as candidates for splitting. Because the regions are treated as separate after splitting, it is possible that homogenous regions of smaller size were enclosed inside bigger regions that were judged non-homogenous. In this case it is possible that two adjacent regions are homogenous, which can lead to over-segmentation. In the second stage of the algorithm, the adjacent homogenous regions are joined to form meaningful objects. The two important issues, which might not be immediately obvious, are: when to stop splitting and when to stop merging the regions back again. Also, it is necessary to define what exactly 'homogeneous' means. In [cortez73] three homogeneity criteria are used. According to the dynamic criterion (5.2), the region is split into two if the difference between the maximum \((G_{MAX})\) and minimum \((G_{MIN})\) greylevel values exceed a threshold \(t_d\). The variance criterion (5.3) dictates a region split if the variance of greylevels in the region is above a threshold of \(t_v\). The criteria for homogeneity are:

\[
\begin{align*}
(G_{MAX}) - (G_{MIN}) &\geq t_d \\
\sigma^2 &\geq t_v \\
|\mu_k - \mu_l| &\leq t_A \text{ for every } k \neq l
\end{align*}
\]
Finally in the average criterion, the split takes place if either of the differences between the average values $\mu_k$ of greyscale levels in resulting regions are higher than $t_A$ (5.4).

In the attempt to establish the best split criterion, merge criterion and in order to find out whether the morphological pre-processing in fact influences the efficiency of the entire split algorithm (i.e. whether it is possible to achieve segmentation into meaningful physical objects or in case of head-and-shoulder techniques: whether the extraction of facial features is possible) a generic algorithm was constructed as is described below. In the first instance the splitting part of the algorithm was tested. The results of the application of variance, average and dynamic criteria were tested on a range of images. The results of tests carried out on the first frame of the Miss America sequence are visualised in (Figure 5.11).

![Figure 5.11: Clockwise from the top left corner: Miss America (original), variance criterion, average criterion and dynamic criterion.](image)

Subsequently the same techniques were applied to images pre-processed using a mathematical morphology based filter. Here, morphological closing followed by morphological opening was applied on 3x3 blocks of pixels. Morphological closing $\gamma^n$ (5.5) and morphological opening $\delta^n$ (5.6) apply morphological erosion $\varepsilon^n$ (5.7) and morphological dilation $\delta^n$ (5.8) in reverse order.
Chapter 5: Fitting of a generic wire-frame model to the scene

\[
y^{(n)}(x, y) = \delta^{(n)}(\epsilon^{(n)}(I)) \quad (5.5)
\]

\[
\varphi^{(n)}(x, y) = \epsilon^{(n)}(\delta^{(n)}(I)) \quad (5.6)
\]

\[
\epsilon^{(n)}(x, y) = \min \{ l(x-\Delta x, y-\Delta y) : -n \leq \Delta x \leq n, -n \leq \Delta y \leq n \} \quad (5.7)
\]

\[
\delta^{(n)}(x, y) = \max \{ l(x-\Delta x, y-\Delta y) : -n \leq \Delta x \leq n, -n \leq \Delta y \leq n \} \quad (5.8)
\]

In all the above equations \(l(x, y)\) denotes the 2D position of the centre of the analysed block (the analysis for each block is the same, the blocks are non-overlapping and cover the entire area of the image), \(n\) is the number of pixels on each side of the centre of the block (i.e. in case of 3x3 blocks \(n = 1\)). The results of a split of the pre-processed image of Miss America are visualised in Figure 5.12. As can be seen, the average criterion results in wrong segmentation in both cases regardless of whether pre-processing was applied or not. The results for the remaining two criteria (the number of squares created at the splitting step) is presented in Table 5.1. The best results (the lowest number of generated squares) was obtained for the case of the dynamic criterion with morphological pre-processing.

Figure 5.12: Clockwise from the top left corner: Miss America pre-processed with morphological opening followed by morphological closing, variance criterion, average criterion and dynamic criterion.
Chapter 5: Fitting of a generic wire-frame model to the scene

Analysis | Number of squares |
--- | --- |
No pre-processing, variance criterion | 7812 |
No pre-processing, dynamic criterion | 8712 |
Morphological filtering, variance criterion | 5544 |
Morphological filtering, dynamic criterion | 5476 |

Table 5.1: Splitting stage results

In order to create physically meaningful regions, the split step must be followed by the merge step. The same homogeneity criteria were used for this purpose. Also, in this case, the dynamic criterion proved to be the most relevant. Also, here, pre-processing using mathematical morphology proved indispensable (Figure 5.13).

![Figure 5.13: Top left: Claire - no pre-processing applied; top right: split-and merge applied to the image on the left; bottom left: pre-processing applied to the top left image, bottom right: split and merge applied to the bottom left image.](image)

As can be seen the oval of the lips was successfully recovered only after application of pre-processing. Unfortunately, these results could not be extended to all the tested images. The
technique fails to localise facial features in the case of Miss America stills. The split-and-merge technique also proved to be very sensitive to values of thresholds (5.2 - 5.4) applied.

It was also noticed that the results obtained vary quite considerably depending on the resolution of the image (Figure 5.14). Since these levels must be chosen heuristically, there is no guarantee that they are optimal, or that they will produce optimal results for other images. Also, there are far more degrees of freedom in ‘adjusting’ the merge algorithm. Since there is usually a number of regions homogeneous to the analysed one, the question arises: should the most homogeneous region only be merged with the analysed one, or should the algorithm take into account the area of the region to be merged. Also, apart from homogeneity, the shape of the regions should be taken into account.

Although segmentation and mathematical morphology seem to be very attractive tools for feature extraction and image simplification respectively, they do not deliver the desired results, i.e. they are not capable of reliable extraction of facial features. The performance is dependent upon settings which require prior training. The fact that the resolution influences the split-and-merge algorithm is very discouraging indeed.

![Figure 5.14: Split and merge technique localises lips at lower resolution (left) but fails to do so if the same algorithm is applied to the full resolution image (right)](image)

5.3 Application of facial data-bases

5.3.1 Introduction

The poor results from mathematical morphology and segmentation from motion techniques provide an important clue, that the feature extraction techniques used traditionally in image and video processing are not necessarily the best tools for extraction of facial features in
head-and-shoulders scenes. The above results, and the considerations that are the subject of Chapter 3, led to the conclusion that an entirely new approach to problems of wire-frame fitting (and tracking) might deliver better results.

In this section new automatic wire-frame fitting algorithms are presented and tested. The common feature of these algorithms is that they are based on a data-base of facial images. The advantage of this approach is that the facial features are, in a sense, part of the processing system a priori. This is consistent with the basics of a knowledge (semantic) based approach, where the approximate contents of the scene are known in advance. All the described algorithms attempt fitting of the Candide [rydfalk87] wire-frame model. More specifically, since there have been several releases of version 1 of Candide, the model consisting of 79 vertices and 108 triangles is used (Figure 5.15 depicts Candide from various angles). This wire-frame was chosen since it produces quite realistic results (Figure 5.16 - Figure 5.19) using a reasonable number of vertices and triangles.

Figure 5.15: Candide model consisting of 79 vertices and 108 triangles
In an effort to derive a reliable technique of automatic wire-frame fitting several algorithms were created and tested. These algorithms and the results of their application to facial images will be presented in the following sub-sections. Finally the comparison between these three will point to the most reliable method.
5.3.2 Correlation based algorithm

First, a method based on correlation was derived and tested. Since the algorithm requires a facial data-base (the terms 'data-base' and 'code-book' will be used equally here), the widely used MIT facial archives [mit] were chosen (Figure 5.20). 64 images were used in tests. These faces were originally used for facial recognition purposes.

Figure 5.20: Example images from the MIT facial archives
Chapter 5: Fitting of a generic wire-frame model to the scene

The proposed algorithm consists of two stages. In the first stage the facial data-base is pre-processed off-line. Once this is done, the on-line processing commences. The pre-processing is applied in order to simplify the contents of the image. Unlike facial recognition systems, the automatic wire-frame fitting algorithm for video coding is not required to choose the image from a data-base which is most similar to the current analysed image. It is therefore possible to reduce the amount of memory necessary to store the data (here: by simplifying each image in the code-book) and still obtain good automatic wire-frame fitting results. This problem becomes relevant in the case of a large data-base. Also, if the system were to be implemented in hardware where the number of memory components must be minimised (e.g. mobile phone application), storage issues must be taken into account.

5.3.2.1 Facial data-base pre-processing

The preparation of the facial data-base is performed only once before the actual processing commences. Therefore, neither the question whether the human operator intervention is necessary or not, nor the timing issues influence further processing. As a matter of fact all the images from the facial code-book are manually fitted with the generic Candide wire-frame model (Figure 5.21).

Figure 5.21: Manual fitting example
The process involves vertex-by-vertex fitting of the wire-frame to the appropriate points on human face. Guidelines similar to those specified in [aizawa89] were followed (section 3.2.1), i.e. the generic wire-frame was first re-sized to the proportions dictated by characteristic points in human face, and then carefully adjusted in the area of important facial features (the eyes, the lips, and the nose - Figure 5.21). The manually fitted wire-frames are used later on as a reference for evaluation of the accuracy of the automatic fit. It is therefore necessary to test the manual fitting by changing action units. The action units used for tests of fitting accuracy are listed in Table 5.2 and their physical interpretation can be viewed in (Figure 5.22 - Figure 5.23). Action units intensity values were chosen so that a significant change in the speaker’s face expression is clearly visible.

<table>
<thead>
<tr>
<th>Action unit number</th>
<th>Action unit intensity</th>
<th>Physical interpretation</th>
</tr>
</thead>
<tbody>
<tr>
<td>au0</td>
<td>0.3</td>
<td>Raise the upper lip</td>
</tr>
<tr>
<td>au1</td>
<td>0.3</td>
<td>Lower the lower lip</td>
</tr>
<tr>
<td>au2</td>
<td>0.5</td>
<td>Stretch the lips</td>
</tr>
<tr>
<td>au3</td>
<td>0.3</td>
<td>Lower the eyebrows</td>
</tr>
<tr>
<td>au4</td>
<td>0.4</td>
<td>Depress the lips’ corners</td>
</tr>
<tr>
<td>au5</td>
<td>0.3</td>
<td>Raise outer part of eyebrows</td>
</tr>
<tr>
<td>au6</td>
<td>1.0</td>
<td>Close upper eyelid</td>
</tr>
<tr>
<td>au7</td>
<td>1.0</td>
<td>Close lower eyelid</td>
</tr>
</tbody>
</table>

Table 5.2: Action units used for testing of the accuracy of fitting

Figure 5.22: Testing manual fitting using specification from Table 5.2 (au0 to au3)
Chapter 5: Fitting of a generic wire-frame model to the scene

The fitted and tested wire-frames constitute a new data-base: the wire-frame data-base. Once the preparation of the wire-frame data-base is completed, all the images in the facial data-base are submitted to pre-processing involving edge detection followed by thresholding. Here an isotropic edge detector was used (Table 5.3) [gonzales77]. Once the thresholded images are created, there is no need to store the original images anymore. Thus, as a result of the pre-processing step, two data-bases are created: a data-base of wire-frames and a data-base of thresholded images. This concludes the first stage of the algorithm.

<table>
<thead>
<tr>
<th>2</th>
<th>2</th>
<th>2</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>-2</td>
<td>-2</td>
<td>-2</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>-2</th>
<th>-2</th>
<th>-2</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>2</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>2</th>
<th>0</th>
<th>-2</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>0</td>
<td>-2</td>
</tr>
<tr>
<td>2</td>
<td>0</td>
<td>-2</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>-2</th>
<th>0</th>
<th>2</th>
</tr>
</thead>
<tbody>
<tr>
<td>-2</td>
<td>0</td>
<td>2</td>
</tr>
<tr>
<td>-2</td>
<td>0</td>
<td>2</td>
</tr>
</tbody>
</table>

Table 5.3: Isotropic templates for edge detector

5.3.2.2 Processing of the unknown image

The unknown image is submitted to the same operations as the images from the code-book (this time on-line) - Figure 5.24. Frame fitting is performed on a coarse-to-accurate basis, i.e. first the system looks for the position of the speaker’s head (coarse stage) and then for the particular important facial features: the left and right eye, the lips and the nose.
Chapter 5: Fitting of a generic wire-frame model to the scene

In the coarse step, from each binary image in the code-book a sub-template containing only the subject's head is extracted (this can be done based on the co-ordinates of the pre-fitted wire-frame). The sub-template is then correlated with the thresholded version of the unknown image. A correlation coefficient \( R_c \) (5.9) is calculated at every possible placement \((m, n)\) of the template \(T\) on the analysed (unknown) image \(I\) (\(L\) and \(K\) are the dimensions of the template and \(M\) and \(N\) are the dimensions of the analysed image).

\[
R_c(m,n) = \sum_{i=0}^{L} \sum_{k=0}^{K} I(m+k, n+l)T(k,l) ; \quad 0 \leq m < M-K \quad 0 \leq n < N-L \tag{5.9}
\]

The correlation process, although performed over the entire image, is relatively fast, since the processed images are by now monochrome bitmaps, and the correlation involves incrementation only. The above analysis is repeated for all the face sub-templates extracted from the code-book images. The correlation peak (maximum value of \( R_c \)) estimates the coarse placement of the wire-frame for a given sub-template. The wire-frame assigned to the sub-template which gave the highest correlation peak is chosen as a rough approximation of the wire-frame of the unknown subject.

As soon as the coarse fit is completed the semantic information can be utilised. Here the information about the geometry of the human face and correlation map are combined in the concept of 'floating templates' (Figure 5.24). The sub-templates of the facial features (eyes, nose and lips - as opposed to sub-templates containing the entire face in the coarse step) extracted from the thresholded images from the code-book are allowed to float over certain restricted areas of the analysed image. The sub-template of the lips is allowed to drift vertically and horizontally in a window with horizontal and vertical dimensions twice those of the lips template itself. Since the size of the sub-template is different for every image from the code-book (its size is deduced from the 2D position of the manually adjusted wire-frame), the size of the area in which the templates are allowed to float will vary from image to image. Similar rules apply to the templates of the eyes and the nose. Similarly to the coarse stage, the final position of the facial feature is defined by the location at which the correlation coefficient reaches its maximum value.
Figure 5.24: On-line processing of the unknown image

Figure 5.25: Vertices describing the left and the right eye

Figure 5.26: Vertices describing the lips and the nose

It is important to note that feature templates from all the images in the codebook are used in the accurate fitting stage - not just those from the 'best fit' stage. This approach was found to
give improved fitting accuracy as might be expected from consideration of the physical character of human faces.

Once both the coarse and accurate fitting steps are completed, the wire-frame of the face in the unknown image is assembled piece-by-piece from the individual wire-frames - which could have been assigned to up to five different subjects. The assembly is performed in the following way. All *Candide* vertices but the ones describing the eyes, the nose and the lips (Figure 5.25 - Figure 5.26) are taken from the wire-frame that was chosen for coarse fit. The vertices corresponding to the eyes are taken from the wire-frame assigned to the code-book image that gave highest eye sub-template correlation peak. The same rules are applied to vertices describing the lips and the nose. Once positioned, the vertices are connected to form the resulting wire-frame of the unknown subject.

### 5.3.2.3 Results of the tests

The algorithm described in the previous paragraphs was tested in the following way. An arbitrarily chosen image was excluded from the MIT facial data-base. The remaining set constitutes the facial code-book. The code-book is subject to the pre-processing presented in section 5.3.2.1 and the excluded image is treated as an unknown and subjected to the on-line analysis described in section 5.3.2.2. The frame fitting is described as successful if after the application of both coarse and accurate steps the facial features are fitted properly with the vertices of the *Candide* model. If at least one of the features has not been fitted properly, the fitting is described as unsuccessful (Figure 5.27 depicts an example of successful and unsuccessful fitting in the area occupied by the lips). The above approach was repeated for all the images from the data-base.

![Figure 5.27: Successful (left) and unsuccessful (right) fitting](image-url)
Although the wire-frame might visually appear to have been fitted properly, it was decided to verify the each successful fit by an application of the action units. Since with knowledge based techniques texture mapping is used to reconstruct the sequence at the receiver side, the application of accuracy measures normally used for evaluation of waveform based algorithms (2.2 - 2.3) seems pointless. High noise figures might be obtained for very good reconstruction results and vice versa: poor reconstruction might deliver low noise figures. Therefore, it was decided to test the accuracy of the automatic fitting algorithm in comparative manner. After the unknown image has been successfully automatically fitted with the Candide wire-frame, the same wire-frame is fitted to the subject manually. Subsequently the same action units and global motion vectors are applied to both subjects. Finally, a short movie with two faces placed side-by-side is created. Only if there are no observable differences in the expressions rendered using the two different wire-frames (one fitted manually and one fitted automatically) is the fit verified as successful. Consider one case of an apparently successful wire-frame fit (Figure 5.28 - right). The result needs to be verified against the manual (i.e. reference) fit. In order to do this, the Candide wire-frame is fitted manually to the same subject (Figure 5.28 - left).

![Figure 5.28: Manual fit (left) and successful automatic fit (right)](image)

The two images are subsequently mapped their respective wire-frames and facial expressions are rendered by activation of action units. Stills from the created movie can be seen in Figure 5.29. The facial expressions on the left side of each sub-figure have been rendered from a manually fitted wire-frame whereas the face expressions on the right from a wire-frame fitted automatically. The action units and global vectors corresponding to each sub-figure are listed in Table 5.4.
Figure 5.29: Testing automatic wire-frame fitting accuracy (LH images: wire-frame fitted manually, RH images: wire-frame fitted automatically)
Chapter 5: Fitting of a generic wire-frame model to the scene

<table>
<thead>
<tr>
<th>Action Unit</th>
<th>Intensity</th>
<th>Sub-figure</th>
</tr>
</thead>
<tbody>
<tr>
<td>Upper lip raiser</td>
<td>+0.5</td>
<td>A</td>
</tr>
<tr>
<td>Lip stretcher</td>
<td>-0.5</td>
<td>B</td>
</tr>
<tr>
<td>Lip stretcher</td>
<td>+0.7</td>
<td>C</td>
</tr>
<tr>
<td>Brow lowerer</td>
<td>+0.3</td>
<td>D</td>
</tr>
<tr>
<td>Lip corner depressor</td>
<td>+0.5</td>
<td>E</td>
</tr>
<tr>
<td>Outer brow raiser</td>
<td>+0.5</td>
<td>F</td>
</tr>
<tr>
<td>Eye closed</td>
<td>+1.0</td>
<td>G</td>
</tr>
<tr>
<td>Lid tightener</td>
<td>+1.0</td>
<td>H</td>
</tr>
<tr>
<td>Lip presser</td>
<td>+0.7</td>
<td>I</td>
</tr>
<tr>
<td>Combined lip stretcher, upper lip raiser</td>
<td>+0.5</td>
<td>J</td>
</tr>
<tr>
<td>lip corner depressor</td>
<td>+0.1</td>
<td></td>
</tr>
<tr>
<td>Rotation about neck axis</td>
<td>-0.4</td>
<td>K</td>
</tr>
<tr>
<td>Rotation about ear axis</td>
<td>+0.2</td>
<td>L</td>
</tr>
</tbody>
</table>

Table 5.4: Action units and global motion vectors used for fitting accuracy testing

Results of the tests carried out on the images from the MIT facial archives were quite encouraging. Apparently successful wire-frame fitting after the second step (facial features fitting) was achieved in 81.3 % of the cases. However some results were not completely satisfactory, i.e. application of action units from Table 5.4 produced unacceptable results. This decreased the successful (and verified) wire-frame fitting rate to 75.0 %.

In most cases where the verification failed this was due to difficulties in fitting of the wire-frame in the area occupied by the speaker’s lips. Visually, the fit appears to be sufficiently accurate but once the action unit responsible for raising upper or lower lip is activated, artefacts - due to the fact that the corner of the lips of the wire-frame does not overlap with the corner of the lips of the subject in the scene - become apparent. In the following subsection a solution to this problem is proposed.
5.3.2.4 An improved algorithm

Since some of the apparently successful fits were not verified to be accurate enough in the area occupied by the speaker's lips, the algorithm proposed in the proceeding sections was extended to add another processing step: a 'dynamic data-base' of features. A dynamic data-base of features can be created during execution of the algorithm, i.e. the new code-book would not take any extra storage space. On the other hand, however, the speed of the fitting algorithm would decrease, since the idea of the extended algorithm includes computationally expensive techniques (e.g. texture mapping etc.). The idea of the improved algorithm is to add another stage to the processing already described. This new stage would be designed to handle very fine fitting of the wire-frame in the area of the speaker's lips. It would target accurate adjustment of the vertices describing the corners of the lips (vertices 31 and 64 in Figure 5.26) since proper fitting would ensure that split between the upper and the lower lip does not lead to disturbing artefacts during reconstruction (Figure 5.30). The improved algorithm description follows.

Once the coarse and accurate stages are completed, (i.e. the best matches for all facial features are found), the algorithm utilises the best-match lips sub-templates further to increase the fitting accuracy. The action units of the wire-frame assigned to the code-book image from which the best-match sub-template was extracted are enabled, so that the number of lip-containing sub-images is dynamically increased to accommodate new shapes (Figure 5.31). The action units applied in order to increase the number of lip sub-templates are listed in Table 5.5 ('lips' dynamic data-base action units').
Chapter 5: Fitting of a generic wire-frame model to the scene

Figure 5.31: Dynamically increasing the number of sub-images containing lip shapes

<table>
<thead>
<tr>
<th>AU</th>
<th>Lower level</th>
<th>Step</th>
<th>Upper Level</th>
</tr>
</thead>
<tbody>
<tr>
<td>Upper lip raiser</td>
<td>0.0</td>
<td>0.1</td>
<td>0.5</td>
</tr>
<tr>
<td>Lower lip lowerer</td>
<td>0.0</td>
<td>0.1</td>
<td>0.5</td>
</tr>
<tr>
<td>Lip corner depressor</td>
<td>-0.5</td>
<td>0.1</td>
<td>0.5</td>
</tr>
<tr>
<td>Lip stretcher</td>
<td>-0.5</td>
<td>0.1</td>
<td>0.5</td>
</tr>
<tr>
<td>Lip presser</td>
<td>0.0</td>
<td>0.1</td>
<td>0.3</td>
</tr>
</tbody>
</table>

Table 5.5: Lips’ dynamic data-base action units

Figure 5.32: The area of adjustment in the third (fine) stage of the algorithm
The new - third stage of the wire-frame fitting (here referred to as the 'fine' stage) takes over where the second (accurate) stage left off, i.e. the initial position of the lips is assumed to be the one established at the end of the accurate stage. At this point the best-match sub-template is again allowed to float vertically and horizontally. However in this case the change in the 2D co-ordinates is limited to one tenth of the horizontal size and one fifth of the vertical size of the sub-template (Figure 5.32).

In reality this results in a relatively small floating region, so that the computational cost involved here is negligible. At each 2D spatial position within the newly created search range the algorithm calculates the correlation coefficient (5.9) in order to establish which of the dynamically created thresholded images of the lips is the most appropriate. Again, since the number of action units chosen to extend the lips' date base is limited, the computational costs of calculating correlation coefficients is low. Once the correlation coefficients for all newly created sub-templates and for all their possible locations within the float region are calculated, the one with the greatest value is chosen. The 2D location at which it was reached is the new location of the centre of the lips. Also, the wire-frame fitted to the subject as a result of the second (accurate) stage of the previous analysis is now modified to accommodate the action unit assigned to the sub-template for which the greatest correlation coefficient was. This concludes analysis of the unknown image in the last (fine) stage of the automatic wire-frame fitting.

5.3.2.5 Results of tests with the improved algorithm

The introduction of the third stage to the original algorithm improved the accuracy of fitting quite considerably. It was possible to verify the fit for all the wire-frames that were judged to have been fitted successfully. Here a movie with two faces placed next to each other was also created for each tested subject in order to verify the accuracy of fitting. A selection of stills using action units from Table 5.4 can be seen in Figure 5.33.

This improvement increased the overall success rate from 75.0 % to 81.25 %. As sometimes it is difficult to visually estimate whether the fit in the lips' area is successful (Figure 5.34), and creation of a separate movie for each tested subject is quite time consuming, a numerical method for evaluation of wire-frame fitting accuracy was developed.
Figure 5.33: Testing automatic wire-frame fitting accuracy after application of the fine fitting step (action units and global motion as in Table 5.4)
The proposed measure takes into account the position of the vertex and its deviation from the reference point (the location of the vertex of the wire-frame fitted manually). The error is calculated as a weighted sum of the partial errors due to inaccurate fitting of particular vertices (5.10).

\[
E_T = \frac{1}{4}(E_{64} + E_{31}) + \frac{1}{8}(E_7 + E_{40}) + \frac{1}{16}(E_8 + E_9 + E_{33} + E_{66}) \tag{5.10}
\]

where \( E_i \) denotes the error of fitting of the \( i \)-th vertex (Figure 5.26). The \( E_i \) errors are calculated in pixels and can be defined as a weighted Euclidean distance between the \( i \)-th vertex fitted manually (reference vertex) \( v_{im} \) and the vertex fitted automatically \( v_{ia} \) (5.11).

(5.10) can be interpreted in the following way: should each of the vertices be displaced by 1 pixel with regard to the reference fit, the total fitting error would be '1', however, an error of half of that value would be obtained should \( v_{64} \) and \( v_{31} \) be displaced by only one pixel (Figure 5.35). Thus (5.10) heavily penalises errors due to inaccurate fitting in the corners of the lips.

In order to find out if the proposed error measure has any physical meaning, the error (5.10) was calculated for the automatically fitted wire-frames before \( (E_{TB}) \) and after application
(\(E_{TA}\)) of the last (fine) stage of the algorithm. In 94% of the cases (Table 5.6) the error decreased to below \(E_{TD} = 3\). In all cases the error decreased. This is to say, that (5.10) presents a sufficiently good measure of the accuracy of fitting in the area occupied by the lips and that in the particular case of the analysed images the threshold value of \(E_T = 3\) might be used to separate verifiable and non-verifiable successful fits.

<table>
<thead>
<tr>
<th>(E_{TD} = E_{TB} - E_{TA})</th>
<th>Fits for which error decreased by more than (E_{TD}) [%]</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.0</td>
<td>100</td>
</tr>
<tr>
<td>3.0</td>
<td>93.8</td>
</tr>
<tr>
<td>5.0</td>
<td>87.5</td>
</tr>
<tr>
<td>7.0</td>
<td>12.5</td>
</tr>
<tr>
<td>8.0</td>
<td>6.3</td>
</tr>
<tr>
<td>9.0</td>
<td>0</td>
</tr>
</tbody>
</table>

Table 5.6: Testing of the error measure

5.3.2.6 Summary

The automatic wire-frame fitting method based on correlation proved to be a relatively reliable tool for fitting of a Candide model. Using the images from the MIT data-base it was possible to fit the model successfully in more than 80% of cases. Facial expressions created by application of action units and texture mapping to an automatically fitted wire-frame were comparable to those created from manually fitted wire-frames. A measure for the error from wire-frame fitting in the area occupied by the lips has been developed. This error measure is sensitive to the situation in which the corners of the lips of the wire-frame are not fitted accurately enough to the corners of the lips of the subject in the scene.

5.3.3 Principal component analysis based algorithm

Although the automatic wire-frame fitting technique based on correlation produced reasonably good results as far as the accuracy of the reconstruction (rendering) is concerned, it seemed that the success rate might be improved if a more application-specific technique was used.
Searching for a more reliable automatic wire-frame fitting technique it was decided to examine what seems to be the most successful mathematical model for facial recognition: the method of principal components. Since the majority of processing in a knowledge based approach concerns faces one might speculate that the approach based on PCA might give better results. Thus an algorithm for automatic wire-frame fitting based on the analysis of principal components of sub-images extracted from the code-book images was investigated. Similarly to the previous case, the initial analysis involved the use of the same 64 images from the MIT facial code book. This ensured a fair comparison with the correlation-based algorithm.

5.3.3.1 Facial data-base preparation

The algorithm is based on the analysis of the principal components of the sub-images extracted from a facial data-base (code-book). Five separate 'sub-sequences' (Figure 5.36) of the sub-images are created. One sub-sequence consists of the sub-images containing entire faces. Each of the remaining four contains sub-images of a different facial feature: the lips, the left and right eye, and the nose. These are referred to as the lips, the left eye, the right eye and the nose sub-sequence respectively.

Figure 5.36: The left eye, the right eye, the nose and the lips sub-sequences
The way each sub-sequence is extracted from the code-book differs considerably from the one used for sub-template extraction in the earlier (correlation) algorithm. As PCA is used in this algorithm, it is relatively important to extract the features’ sub-sequences with reference to a certain characteristic point of the given feature (or entire face). In the case of the eyes the lips and the nose, the reference extraction points are compatible to those used in the PCA-based tracking algorithm described and successfully tested in the previous chapter.

The reference point for the face sub-sequence was chosen to be the mid-point between the centres of the left eye corona and the right eye corona (Figure 5.37).

Since (similarly to the previously described automatic model fitting scheme) all the faces from the code-book of facial images must be manually pre-fitted with the Candide wire-frame, where possible, the co-ordinates of the vertices of the wire-frame were utilised, e.g. in the case of the lips the reference point was calculated using the co-ordinates of vertex 64 and 31 (Figure 5.26). The reference points for the eyes were chosen manually (the co-ordinates of the manually fitted Candide were not utilised). Once all the sub-images for a given sub-sequence are successfully extracted, they are converted into 1D vectors by concatenating subsequent lines of the sub-image (Figure 4.4). This process yields five (two eyes, nose,
Chapter 5: Fitting of a generic wire-frame model to the scene

mouth and face) sets of input vectors (in the sense defined by equation 4.1). However, it should be noted here, that the value of $M$ defined in (4.1) as the number of initial frames, has an entirely different meaning here. Now, $M$ is the number of images in the facial data-base. In order to avoid confusion, the number of images in the facial code-book will be referred to as $M_F$. Unless otherwise stated in this chapter, any reference to the equations describing the principal component space assumes that its dimension is $M_F$ rather than $M$.

OFF-LINE FACIAL DATA-BASE PREPARATION

Figure 5.38: PCA-based wire-frame fitting facial data-base preparation

All the sub-images are normalised using the expected value (mean) of a particular sub-sequence (4.2). This is followed by calculation of a principal components space for each extracted, scanned and normalised sequence of sub-images (4.3) - (4.12). At this point, the pre-processed system consists of six separate data-bases (all stemming from a facial code book): the data-base of wire-frames, principal components space of faces, and four principal component spaces describing each of the important facial features (Figure 5.38). The entire processes described above is performed off-line, i.e. as a part of the system preparation. It does not influence the speed of the on-line processing, which is performed without human intervention. In a real-life situation one would expect the results of the off-line step to be stored in the memory of both transmitter and receiver as the a priori knowledge. Since the data-base of pre-fitted wire-frames from the correlation-based approach was used, the figures (Figure 5.20 - Figure 5.23) demonstrating manual fitting examples, and Table 5.2 describing action units used to test the accuracy of manual fitting, are also relevant for the PCA-based approach.
5.3.3.2 On-line algorithm

As in the case of correlation-based approach, the fitting of the unknown image is performed in two stages (Figure 5.39). In the first stage (the coarse stage) the approximate position of the subject’s face is established. This is accomplished by the analysis of the principal components space of the sequence of sub-images containing faces (the face sequence) extracted from the facial code-book.

The analysis in the coarse stage is performed as follows. A sub-image of the same dimensions as the images from the face sequence is extracted at every possible location of the unknown image. It is subsequently converted into a 1D column vector by scanning the image line by line, and then projected onto the principal components space of the face sequence. For this purpose (4.12) is used, obviously remembering that in this case $u_i$ is the $i$-th eigenvector of the principal component space created by the faces extracted in the off-line step (not an image from the initial set), $m$ - the expected value of the face sub-sequence, and $x_1$ the sub-image extracted from the currently analysed unknown image. Since the analysed principal components space was created using face sub-images only, the projection of an unknown image tells us how similar the unknown sub-image is to all the images from the face sequence. Or, in other words, it allows us to judge whether the analysed (unknown) sub-image is a face or not. As before, this can be quantitatively described by the distance described by (4.14). This distance is calculated for each sub-image extracted from the unknown image. The spatial location at which (4.14) applied to the extracted sub-image reaches a minimum is the approximate position of the face on the unknown image and is referred to as the ‘best match face location’ (Figure 5.39).

Once the coarse position of the object’s face is estimated, it is still necessary to choose the wire-frame that would be fitted to the unknown image. As was mentioned before, the Candide wire-frame had been pre-fitted to all the images from the facial code-book. On the other hand, the distance (4.14) does not refer to any specific sub-image from the face sequence. In order to find out which wire-frame is most appropriate at the best match location the distance measure is defined by (4.13). Again, the meaning of the $a$ and $b$ vectors is different here; $a$ is the projection of the sub-image extracted from the unknown image at the best match location and $b_j$ is the projection of the $j$-th image from the face sequence onto principal components space of the face sequence. Again the minimum value of (4.13) tells us which wire-frame to use (index $j$) for coarse fit at the best match location. Once both the best
match location and the wire-frame are established, the first stage (the coarse stage) of the algorithm is concluded (Figure 5.39).

In the second stage of the algorithm both the results from the first stage and the information about the geometry of the human face are combined in order to achieve faster and more reliable operation in a similar manner to that proposed for second step of the correlation-based algorithm (Figure 5.24).

In the second stage of the PCA-based approach the principal components space of the sequence of sub-images containing important facial features (the left eye sequence, the right eye sequence, the nose sequence and the lips sequence) is examined. Since the principal component space of the faces is not used in any further part of the algorithm, all the reference to the analysis of principal components described in greater details for the tracking algorithm (previous chapter) now concerns the principal component spaces derived from the subsequences of facial features. The method of accurate fitting in stage two will be explained from the example of the left eye sequence. A sub-image of the same dimensions as the
images from the left eye sequence is extracted at every possible location of a search region surrounding the given facial feature. The search region is centred on the coarse position of the pupil of the left eye (as indicated after completion of the coarse stage). The dimensions of the search region are adjustable, but in this case the dimensions of the sub-image from the left eye sequence are simply doubled in order to obtain the dimensions of the search region. The further analysis is similar to that carried out in the coarse stage: the best match location of the left eye of the subject is determined using (4.14). This is followed by a search for the most appropriate wire-frame of the left eye using distance (4.13). The vertices corresponding to the left eye (Figure 5.25 - left) are subsequently extracted from this wire-frame and fitted (at the best match location of the left eye) to the wire-frame chosen in the coarse stage - in the same way as was done in the correlation-based algorithm. The above procedure is repeated for the right eye, the nose and the lips. Thus, again the final wire-frame is assembled of five partial wire-frames: the wire-frame of the face (found in the coarse stage), and four wire-frames of the important facial features (found in the second stage). This concludes the automatic wire-frame fitting algorithm.

5.3.3.3 Results of the tests and summary

In order to be able to compare the PCA and correlation-based approaches, the tests were carried out on the images from the same facial data-base (MIT facial archives). In these tests successful and verified wire-frame fitting rate was 87.5 %. The verification was carried out, as on previous occasions, using the action units and global motion specified in Table 5.4, (Figure 5.40 - Figure 5.41).

These results prove, that the PCA-based algorithm is an improvement on the correlation-based method. Although this fact alone would be sufficient to claim that the PCA-based method is better, the analysis of principal components also proves to be faster - thanks to the application of SVD. While wire-frame fitting using PCA took an average of 35.3 sec (on the subjects from MIT facial data-base), it took an average of 83.1 sec to fit the wire-frame using correlation-based approach.

The search for facial features is normally limited to a selected region (this was the case in both PCA and correlation based methods). This is to reduce computational costs and the possibility of finding a wrong match somewhere else in the image.
Figure 5.40: Testing the accuracy of automatic fitting using PCA-based method, action units as in Table 5.4
Figure 5.41: Testing the accuracy of automatic fitting using PCA-based method in the presence of facial occlusions (moustache, beard), action units as in Table 5.4
In order to provide additional proof of the superiority of the method of principal components, the facial features search regions were extended to include the rest of the image (i.e. the speaker and the background). Since the images of the left and the right eye leave the most distinct signature (due to its distinct combination of dark and bright pixels - both in the principal component space and in the image space), it was these two features that were the subject of the tests. Ideally the best matches should remain where they were. Obviously, this proved not to be the case in all the experiments. Global searches performed using PCA confirmed 81.2% of matches. The same experiment for the correlation method gave less than 50% correct matches. This proves that the PCA based method is relatively robust against erroneous feature identification, something that cannot be said about the correlation based approach.

5.4 Summary

In this chapter methods of automatic wire-frame fitting have been proposed and tested. It has been established that the method based on principal component analysis performs with the greatest reliability and should be used in further research. The above result may be explained by the fact that in principal component analysis the most characteristic features of the analysed image are taken into account in the first place. This allows precise localisation of specific patterns (for example: the eye, where distinct combination of dark and light pixels allows reliable recovery) in the image.

It also seems that the idea of application of a facial data-base for the purpose of wire-frame fitting was of great value for accurate fitting. The issues of facial recognition (normally a machine vision technique) and extremely low bit-rate coding (an image processing technique) are normally rigorously separated, but this work has proved that the two areas are closely linked and have to be considered together for a successful solution of the problem of model fitting in knowledge based moving image coding.

Since in model based coding the scene is viewed as a 2D projection of a 3D real-life scene, the application of error measures like MSD (Mean Square Difference) or MAD (Mean Absolute Difference) - commonly used in processing of 2D waveforms - is not very applicable. Therefore the automatic wire-frame fitting quality has been tested subjectively against fitting performed manually.
6.1 Introduction

In Chapter 4 a method of automatic tracking of the facial features for model-based coding was proposed. The tracking was initialised by manual extraction of the facial features from a set of starting frames. In Chapter 5, a method of automatic frame fitting by extraction of facial features was proposed.

In this chapter a unified approach to the problems of model fitting and model tracking in head-and-shoulders scenes is described. An algorithm for tracking the facial features without manual intervention is presented. The manual initialisation described in Chapter 4 is replaced by an automatic one based on the best performing algorithm presented in Chapter 5.

Since the comparison between the fitting approaches in Chapter 5 proved that the method based on the analysis of principal components of the sequence of sub-images suits best the purpose of automatic fitting, this method will be adapted. Although no human intervention is assumed in the on-line algorithm (i.e. extraction of the sub-sequences from initial frames of the analysed sequence is assumed to be automatic), the preparation of the system may still include manual steps. It should, however, be noted, that the off-line step (i.e. preparational step) would normally take place at the time of manufacturing of the system. Thus, the facial database, the data-bases of features and the data-base of adjusted wire-frames would be stored in the transmitter and the receiver by the manufacturer. Since the user does not have to perform any data-base preparation, the system can be fairly described as automatic.

Since a simple fusion of the algorithms presented in Chapters 4 and 5 was not possible, certain additional techniques were used. The application of greyscale linearisation in both the off-line and the on-line stages allowed integration of otherwise two separate algorithms. Although histogram equalisation is first applied in preparation of the generic data-base (Section 6.2.1), it is discussed in detail later in this chapter (Section 6.2.2), since it was the
experience gathered during construction of the on-line algorithm that led to application of this technique both in the on-line and off-line stages.

6.2 Algorithm description

The framework of the proposed automatic fitting/tracking algorithm is presented in Figure 6.1. The off-line stage (generic data-base preparation) consists of spatial equalisation, greyscale equalisation and principal component analysis and is similar to the off-line stage of the algorithm presented earlier in Chapter 5 (Section 5.3.3.1). The on-line processing stage was sub-divided into initial tracking and standard tracking. The initial tracking is performed on $S$ frames (starting with frame 0 of the analysed sequence). The standard tracking algorithm takes over after initial tracking in $S$ frames is completed. During the initial tracking, a 2-stage search, based on the on-line part of the technique described in Chapter 5 (Section 5.3.3.2), is performed on each frame in order to localise the important facial features. Finally, for standard tracking, the technique described in Chapter 4 is applied.

All the elements of the off-line stage (generic data-base preparation) and the on-line stage (processing of a video sequence) are described in detail in the following sections.

![Figure 6.1: Framework of an automatic fitting/tracking algorithm](image-url)
6.2.1 Generic data-base preparation

Since the data-base of the facial images used in processing must be sufficiently representative, a range of images was chosen from various sources. These include the MIT facial data-base, the Manchester facial data-base [peipa] and other independently acquired images Figure 6.2. A total of 64 subjects were used, out of which 25% were female. Before any pre-processing was applied, the images were re-sized, so that faces and facial features in all of them were of the same scale.

![Figure 6.2: Imagery used to construct the facial data-base](image)

This was followed by manual extraction of the facial features (fitting of semantic wire-frame) to the images that had not been previously used in analysis. Since tracking of important facial features was targeted, this included manual extraction of the centres and boundaries of facial features and centres and boundaries of faces. The guidelines first proposed in [aizawa89] (Figure 3.2 - Figure 3.3) were also applied here (Figure 6.3 - Figure 6.4). Since the dimensions of the face to extract are not fully defined in Aizawa's work, the location of the last face boundary extraction point (marked with a black cross in Figure 6.3) must be deduced using proportions of the universal model of the face.
6.2.1.1 Spatial equalisation

Once the features were extracted, the locations and dimensions of the features were stored in a data-base containing the 2D co-ordinates of the reference point and distances to the upper, the lower, the left and the right edges of the feature. The same approach was used for face extraction.

Since the principal component analysis requires that all the sub-images in a given code-book (i.e. code-book of faces, code-book of the left eye, etc.) are of the same dimensions, the largest distances from the reference point to the feature (face) boundaries were chosen in a process called 'spatial equalisation'. This process of establishing the spatial dimensions of sub-images can be summarised in the following way.
Five points are extracted manually from each image containing face (Figure 6.3) or facial feature (Figure 6.4). These five points describe the dimensions of a particular sub-image and the reference point of the face or the feature (e.g. in the case of the left eye, the reference point would be the centre of the corona of the eye). Let us consider a sequence of sub-images containing lips (the process is carried out in the same way for the sub-sequence of faces and the sub-sequences of the remaining facial features).

Since the features were extracted manually, all the images in the sub-image sequence before spatial equalisation may have different dimensions. Suppose there were only two sub-images in the sequence: A and B - both containing lips. If for the sub-image A the distance from the lips' reference point to the lips' left boundary was $a_1$, and for another sub-image B, the same distance was $b_1$, where $a_1 < b_1$, then the distance to the left boundary of the lips for all the resulting images in the sequence (denoted 'R' in Figure 6.5) of sub-images containing lips would be set to $b_1$. Similar rules are applied to the remaining distances. In the presented example (Figure 6.5) $a_1 < b_1$, $a_2 > b_2$, $a_3 > b_3$, and $a_4 < b_4$ thus the distances of the sub-images in the sequence would be set to $b_1$, $a_2$, $a_3$ and $a_4$. In practical case (i.e. when there are more than two sub-images in the sequence), the distances 1-4 would be increased to match the largest corresponding distances in the analysed sequence of sub-images.

Once the facial data-base is annotated using the guidelines presented above, five sequences of sub-images are extracted from all its elements (similarly to the algorithm described in Chapter 3, they will be referred to as 'sub-sequences'). Thus five sub-sequences are created: one sub-sequence of faces, and four sub-sequences of facial features.
6.2.1.2 Greyscale equalisation (data-base preparation)

Although the pre-processing described above provides harmonisation of the spatial resolution for each sub-sequence, the greyscales of pictures taken under different lightning conditions can vary quite substantially, effectively disqualifying the data-base as a source of comparable images. In order to reduce or even eliminate the effect of varying lightning conditions, each sub-sequence was subjected to a histogram equalisation process. Even though histogram equalisation performed on an entire image enhances the details in the human face (Figure 6.6), this is still not sufficient for reliable extraction of facial features.

Figure 6.6: Images before (left) and after (right) histogram equalisation
In order to sufficiently enhance the areas occupied by the facial features, histogram equalisation should be applied after extraction of the particular sub-image, e.g. a sub-image containing the eye is equalised using the greyscales enclosed in the eye’s template only. This approach ('local equalisation') allows the system to obtain sub-images of much better quality (Figure 6.7).

Once spatial equalisation and local histogram equalisation are completed, the code-book is subjected to principal component analysis. The principal component analysis is performed separately on each sub-sequence yielding five separate principal component spaces: one of the sub-sequence of faces and four for the sub-sequences of facial features (Figure 6.8). In practice, only the eigenvectors responsible for the creation of each of the principal components spaces and the spatial locations of the centres of the particular facial features

Figure 6.7 Global and local histogram equalisation
must be stored. This would reduce the storage requirement should a hardware implementation be required.

Figure 6.8: Pre-processing prior to application of the principal component analysis

6.2.2 On line processing of a video sequence

Since very encouraging results, as described in Chapter 4, were obtained using the principal component spaces of facial features extracted from the initial frames of the sequence, the algorithm presented in this section tries to adapt the previously developed technique to the new situation, i.e. where the initial extraction is performed automatically. Because of this the tracking algorithm as such can be sub-divided into two logical parts: initial tracking, and standard tracking. In the initial tracking the data-base of facial images (pre-processed in the way described earlier in this chapter) is utilised. It becomes redundant once the algorithm switches to standard tracking mode. Since the application of histogram equalisation (Section 6.2.1.2) was not an obvious choice during construction of this algorithm, the issues concerning equalising the levels of greyscale are discussed in more detail in section 6.2.2.1.
6.2.2.1 Initial tracking

During initial tracking the system attempts recovery of important facial features using the 2-step algorithm described in section 5.3.3 based on principal components with the following modifications: the unknown image is one of the initial frames of the sequence (Figure 6.9), and the data-base of images is pre-processed in the way described in the previous section.

In the first instance the algorithm seeks the face in the scene. For this purpose the principal components space of the sub-sequence of faces is utilised (Figure 5.36, Figure 5.39). Once the approximate position of the head is found, the algorithm seeks for particular facial features - now utilising principal components spaces of the left eye, the right eye, the nose, and the lips. This second step, again, is performed on an area surrounding the approximate location of a given facial feature (Figure 5.32). However, in both cases the extraction of the sub-image from the unknown image (initial frame of the analysed sequence) is immediately
followed by histogram equalisation (Figure 6.10). Only then is the unknown image projected onto the principal component space of the appropriate feature (or face).

The same algorithm (coarse-to-accurate feature extraction with application of histogram equalisation) is repeated for \( S \) initial frames of the analysed sequence (Figure 6.9). The algorithm is applied in intra mode, i.e. the information about the positions of facial features in one frame is not utilised for the search in the following or proceeding frames.

In the previous experiments with automatic wire-frame fitting using principal components analysis it was established that 87.5% of automatic fitting efforts were successful. Anticipating a lower yield in the case when a generic data-base is applied, the following method of elimination of erroneous results elimination was applied. The number of the initial frames was set to \( S = 1.5 M \), where \( M \), according to the notation used in (4.2) (and the entire manually initialised automatic tracking algorithm) denotes the number of initial set sub-images (not to be confused with number of initial frames \( S \) first introduced in this paragraph). The features were therefore searched for in the initial \( S = 1.5 M \) frames. The results for each feature were subsequently arranged in order of decreasing value of (4.14) (the distance measure used to localise the best match location for a given feature). 30% of the results with the lowest ‘best matches’ were simply discarded. This leaves \( S' = M \) initial frames, which are likely to have had the facial features properly extracted. This approach was used as an alternative to setting up a threshold for the value of (4.14). Although less straightforward and more time consuming (because of the necessity to calculate the locations of the facial features in a greater number of initial frames) this method does not require any ‘training’ for the threshold that would allow the outcome of (4.14) to be immediately classified as a ‘hit’ or ‘miss’. Once \( M \) of the \( S \) initial frames are selected as those with properly extracted facial features, they are used in the second: standard tracking stage starting with frame \( S+1 \).

6.2.2.2 Greyscale equalisation (initial tracking)

Since the histogram equalisation is a new element in the algorithm described in this chapter the reason for its use deserves explanation. The histogram of a typical scene that has been linearly quantised is usually skewed towards the darker levels (Figure 6.11). The function \( h(l) \) describes how many pixels in the analysed image have greyscale level \( l \). It is the task of histogram equalisation technique to transform the \( h(l) \) curve into a straight line. Because of this the histogram equalisation technique is also referred to as ‘histogram linearisation’.
Chapter 6: Fitting and tracking: a unified approach

Figure 6.11: Usual shape of histogram (left) and its linearised version (right)

In practical applications, perfect histogram equalisation is not possible, since the $h(l)$ is discrete, not continuous (Figure 6.12). Although the equalised version of the discrete image histogram is hardly linear, the new distribution of the greyscales enhances the analysed image thanks to more linear distribution of greyscales in the histogram.

Figure 6.12: Discrete (8 level, 64 x 64 pixels) histogram before (left) and after equalisation (right)
Figure 6.13: Initial tracking of Claire (frame 0) without (left) and with (right) equalisation

Figure 6.14: Initial tracking of Claire (frame 5) without (left) and with (right) equalisation

Figure 6.15: Initial tracking of Claire (frame 10) without (left) and with (right) equalisation
Figure 6.16: Initial tracking of Salesman (frame 0) without (left) and with (right) equalisation

Figure 6.17: Initial tracking of Salesman (frame 5) without (left) and with (right) equalisation

Figure 6.18: Initial tracking of Salesman (frame 10) without (left) and with (right) equalisation
As has been emphasised in this chapter, greyscale (histogram) equalisation steps are applied both in the data-base preparation (Section 6.2.1.2) and initial tracking (Section 6.2.2.1) steps. In both cases equalisation is applied locally, i.e. only the sub-images (not entire images) are subject to this process.

In early stages of the development of this algorithm, because of reliable operation of earlier developed algorithms, the use of histogram equalisation was not anticipated (neither in database preparation nor in initial tracking), i.e. the algorithms described in Chapters 4 and 5 were merged without any modifications. This however delivered very disappointing results (Figure 6.13 - Figure 6.18, images on the left). In most cases the features were erroneously located. The most common mistakes were made in localisation of the eyes where, in many cases, the region between the eyebrow and the upper eye-lid was interpreted as the centre of the eye. In some cases the error in localisation of the facial features was inherent to erroneously localised speaker’s head (e.g. Figure 6.14). This can be explained by the fact, that some of the images from the generic data-base (especially those taken from the Manchester facial data-base) were of particularly poor quality (i.e. low contrast). Application of histogram equalisation in both preparational and initial tracking steps modifies the histograms of both the images from initial set and the images from the extracted set (in the sense described in Chapter 4) so that the images from the data-base and the images from the analysed sequence have comparable distributions of greyscales. The above hypothesis was confirmed after carrying out experiments on sequences listed in Table 4.2. Results of these tests are depicted on the example of the Claire and Salesman sequences (Figure 6.13 - Figure 6.18, images on the right).

6.2.2.3 Standard tracking

Once the initial tracking is completed, the principal component space of faces is not used for the purpose of tracking the motion of facial features. This leaves the algorithm with the requirement of independent tracking of four facial features based on four independent principal component spaces.

Since the method for automatic tracking with initial human intervention has already been developed and tested with very good results (as presented in Chapter 4) it seemed reasonable to adapt the existing algorithm to the new needs: i.e. to the modify it so that no human intervention is necessary. Also, application of the tracking algorithm from Chapter 4 reduces
computational costs (the initial tracking step involves two-stage global search, whereas in the automatic tracking method presented in Chapter 4 only images from a relatively small search region are analysed). For this reason, the principal component spaces calculated using the generic data-base are now discarded. Starting from frame $S+1$ they are replaced by four principal component spaces (one for each feature) calculated using the positions of the facial features established in $M$ (out of $S$) initial frames of the sequence. Once this is done, the tracking is carried out exactly as described in Chapter 4 (Figure 6.19).

Figure 6.19: Principal component data-bases are changed after $S$ frames

In contrast to both the preparational step (off-line processing of a generic facial data-base) and the initial tracking, the histogram equalisation process is not applied during standard tracking. This allows application of the successful algorithm from Chapter 4 without any changes and decreases computational costs.

6.3 Results of the tests

The results of the tests carried out on widely used head-and-shoulders sequences Table 4.2 are presented in the form of a mean error and standard deviation (Table 6.1 - Table 6.3). Again, because of the length of the sequences, a representative sample of sequence frames with manually extracted 2D locations of the facial features was used (Table 4.3). These results for automatic fitting and tracking correspond to the ones obtained in Chapter 4 (Table 4.4 - Table 4.6).
Chapter 6: Fitting and tracking: a unified approach

Figure 6.20: Claire (frame 93) manual (left) and automatic (right) initialisation

Figure 6.21: Claire (frame 102) manual (left) and automatic (right) initialisation

Figure 6.22: Claire (frame 104) manual (left) and automatic (right) initialisation
Figure 6.23: *Salesman* (frame 202) manual (left) and automatic (right) initialisation

Figure 6.24: *Salesman* (frame 290) manual (left) and automatic (right) initialisation

Figure 6.25: *Salesman* (frame 300) manual (left) and automatic (right) initialisation
**Chapter 6: Fitting and tracking: a unified approach**

| Facial feature | Miss America | | Claire | |
|----------------|-------------|--------|--------|
| Left eye | 0.74 | 0.62 | 0.64 | 0.60 |
| Right eye | 0.92 | 0.75 | 1.05 | 0.77 |
| Nose | 0.41 | 0.37 | 0.78 | 0.35 |
| Lips | 1.02 | 0.72 | 1.11 | 0.92 |

Table 6.1: Tracking error results for *Miss America* and *Claire*: automatic initialisation

| Facial feature | Car Phone | | Grandma | |
|----------------|-----------|--------|--------|
| Left eye | 1.11 | 0.92 | 0.76 | 0.40 |
| Right eye | 1.31 | 0.90 | 0.83 | 0.73 |
| Nose | 1.51 | 1.00 | 0.71 | 0.61 |
| Lips | 1.91 | 1.13 | 0.87 | 0.79 |

Table 6.2: Tracking error results for *Car Phone* and *Grandma*: automatic initialisation

| Facial feature | Salesman | | Trevor | |
|----------------|----------|--------|--------|
| Left eye | 0.86 | 0.82 | 0.84 | 0.75 |
| Right eye | 1.03 | 0.99 | 0.70 | 0.70 |
| Nose | 0.75 | 0.70 | 0.85 | 0.70 |
| Lips | 1.31 | 1.28 | 1.02 | 0.73 |

Table 6.3: Tracking error results for *Salesman* and *Trevor*: automatic initialisation
Chapter 6: Fitting and tracking: a unified approach

Left eye track [%]  
<table>
<thead>
<tr>
<th>Name</th>
<th>V. good</th>
<th>Good</th>
<th>Satisfactory</th>
</tr>
</thead>
<tbody>
<tr>
<td>Miss America</td>
<td>90.0</td>
<td>10.0</td>
<td>0.0</td>
</tr>
<tr>
<td>Claire</td>
<td>88.2</td>
<td>8.8</td>
<td>3.0</td>
</tr>
<tr>
<td>Car Phone</td>
<td>56.3</td>
<td>41.2</td>
<td>2.5</td>
</tr>
<tr>
<td>Grandma</td>
<td>96.1</td>
<td>3.9</td>
<td>0.0</td>
</tr>
<tr>
<td>Salesman</td>
<td>55.0</td>
<td>40.0</td>
<td>5.0</td>
</tr>
<tr>
<td>Trevor</td>
<td>80.0</td>
<td>20.0</td>
<td>0.0</td>
</tr>
</tbody>
</table>

Right eye track [%]  
<table>
<thead>
<tr>
<th>Name</th>
<th>V. good</th>
<th>Good</th>
<th>Satisfactory</th>
</tr>
</thead>
<tbody>
<tr>
<td>Miss America</td>
<td>90.0</td>
<td>10.0</td>
<td>0.0</td>
</tr>
<tr>
<td>Claire</td>
<td>79.0</td>
<td>21.0</td>
<td>0.0</td>
</tr>
<tr>
<td>Car Phone</td>
<td>68.8</td>
<td>21.2</td>
<td>10.0</td>
</tr>
<tr>
<td>Grandma</td>
<td>96.1</td>
<td>3.9</td>
<td>0.0</td>
</tr>
<tr>
<td>Salesman</td>
<td>65.0</td>
<td>30.0</td>
<td>5.0</td>
</tr>
<tr>
<td>Trevor</td>
<td>80.0</td>
<td>20.0</td>
<td>0.0</td>
</tr>
</tbody>
</table>

Table 6.4: The left and the right eye tracking quality: automatic initialisation

Nose track [%]  
<table>
<thead>
<tr>
<th>Name</th>
<th>V. good</th>
<th>Good</th>
<th>Satisfactory</th>
</tr>
</thead>
<tbody>
<tr>
<td>Miss America</td>
<td>100.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>Claire</td>
<td>88.2</td>
<td>11.8</td>
<td>0.0</td>
</tr>
<tr>
<td>Car Phone</td>
<td>60.0</td>
<td>16.2</td>
<td>23.8</td>
</tr>
<tr>
<td>Grandma</td>
<td>89.6</td>
<td>10.4</td>
<td>0.0</td>
</tr>
<tr>
<td>Salesman</td>
<td>52.5</td>
<td>47.5</td>
<td>0.0</td>
</tr>
<tr>
<td>Trevor</td>
<td>90.0</td>
<td>10.00</td>
<td>0.00</td>
</tr>
</tbody>
</table>

Lips track [%]  
<table>
<thead>
<tr>
<th>Name</th>
<th>V. good</th>
<th>Good</th>
<th>Satisfactory</th>
</tr>
</thead>
<tbody>
<tr>
<td>Miss America</td>
<td>97.0</td>
<td>3.0</td>
<td>0.0</td>
</tr>
<tr>
<td>Claire</td>
<td>58.8</td>
<td>41.2</td>
<td>0.0</td>
</tr>
<tr>
<td>Car Phone</td>
<td>32.5</td>
<td>40.0</td>
<td>27.5</td>
</tr>
<tr>
<td>Grandma</td>
<td>89.6</td>
<td>10.4</td>
<td>0.0</td>
</tr>
<tr>
<td>Salesman</td>
<td>42.5</td>
<td>36.3</td>
<td>21.2</td>
</tr>
<tr>
<td>Trevor</td>
<td>70.0</td>
<td>30.00</td>
<td>0.00</td>
</tr>
</tbody>
</table>

Table 6.5: The nose and the lips tracking quality: automatic initialisation

A comparison of the results obtained using manual and automatic feature extraction at the beginning of the sequence suggests that the manual method should be slightly more accurate. However, no subjective indication of this fact was visible on tested movies. The most radical differences in tracking accuracy identified during frame-by-frame examinations of manually and automatically initialised algorithms are presented in (Figure 6.20 - Figure 6.25). As in the manually initialised case, tracking was maintained for all the features and for all the sequences. The fact that in most of the cases, the mean error is still less than one pixel, confirms that the tracking using automatic initialisation was successful.

In 37.5 % of cases (as can be seen from results comparison: Table 4.4 - Table 4.6 and Table 6.1 - Table 6.3) the tracking using automatic initialisation was no worse than that using manual extraction of the important facial features. At this point it is worth observing that no
filters were applied during the tracking. It is possible that the results might be improved by the application of 'smoothing' algorithms. However, the technique is more attractive without the application of any 'optimising' aspects. Also, knowledge about the structure of the face was not explicitly used in the tracking. i.e. it was not necessary to use the tracking path of e.g. the left eye in order to recover the position of the right eye. The features used their own principal component spaces with sufficient consistency to keep the tracking while maintaining low levels of error.

The facial features' tracking quality can be also quantified by analysis of the normalised error distance (NED), using the same criteria as in Chapter 4 (Table 6.4 - Table 6.5). The results prove that tracking is of very good quality in more than 50% frames in all but one case (tracking the lips in Car Phone sequence) - Table 6.4 - Table 6.5.

Also, since principal component analysis is a highly regular method of analysis of a sequence of input vectors, a hardware implementation might be simplified even further. This is not to say that the implementation of a software codec in real time is not possible. The tests were carried out on rather slow machine (120MHz Pentium-based PC computer) and no code optimisations were made. The tracking performance is summarised in Table 6.6. As can be seen, at the present moment (April, 1998), the algorithm is capable of processing of 3.4 frames per second per one facial feature.

<table>
<thead>
<tr>
<th>Sequence</th>
<th>Total execution time [s]</th>
<th>Execution time/frame [s]</th>
</tr>
</thead>
<tbody>
<tr>
<td>Miss America</td>
<td>44.3</td>
<td>0.30</td>
</tr>
<tr>
<td>Claire</td>
<td>50.1</td>
<td>0.30</td>
</tr>
<tr>
<td>Grandma</td>
<td>220.7</td>
<td>0.29</td>
</tr>
<tr>
<td>Phone</td>
<td>114.9</td>
<td>0.29</td>
</tr>
<tr>
<td>Salesman</td>
<td>119.3</td>
<td>0.30</td>
</tr>
<tr>
<td>Trevor</td>
<td>29.4</td>
<td>0.29</td>
</tr>
</tbody>
</table>

Table 6.6: Execution times for tested sequences

6.4 Summary

The construction of an algorithm capable of fitting and tracking of the vertices for 'head-and-shoulders' scenes proved to be successful. The unified approach to the problems of fitting
and tracking presented here, produces consistent and robust results as demonstrated, both subjectively (by analysis of the video sequences) and objectively by analysis of the mean error, standard deviation and normalised error distance.

The algorithm is based on two separately-developed techniques: one for automatic fitting of the model without tracking, the other for automatic tracking, with manually extracted facial features in the initial frames of the sequence. Since both fitting and tracking utilise a common mathematical background, it is easier to apply the algorithm in practice. The automatic fitting/tracking algorithm consists of two stages: off-line data-base preparation and on-line tracking. The new element in both on-line and off-line stages is application of histogram equalisation (linearisation). In the off-line stage, the histogram linearisation is applied locally after spatial equalisation in order to provide a uniform data-base of features and faces. In the on-line stage, the histogram linearisation is only applied in the initial tracking (tracking of initial frames). Tracking in the remaining part of the sequence (standard tracking) does not require application of histogram equalisation.

Fusion of the two methods presented in Chapters 4 and 5 allows elimination of the human factor from the processing, i.e. on-line processing is carried out without intervention of the system operator.

The application of fully automatic initialisation results in larger overall tracking errors. This fact is reflected in a larger number of frames in which tracking can be described as good, or satisfactory, rather than very good. Despite larger mean errors, tracking was maintained in all sequences for all facial features.

The method has been tested on widely used ‘head-and-shoulders’ scenes and achieved reliable tracking in all cases. The performance was only marginally poorer than the manually initialised approach. The achievement of automatic fitting and tracking in a head-and-shoulders scenes represents a significant development in the quest for an extremely low bit rate video encoder system.
Chapter 7

Conclusions and future research

7.1 Introduction

Useable video communication over extremely low bandwidth channels (below 10 kbit/s) has for years been unobtainable. Simple modification of the existing waveform-based algorithms is insufficient to provide an acceptable quality of moving image. Whereas the waveform-based approach seems to provide sufficient quality video at bitrates above 64 kbit/s, the unacceptable compression artefacts become only too visible at extremely low bitrates. A technique more suited to low bit rate applications in therefore required.

In this thesis, research into the most important issues concerning semantic based video coding - a technique that has a potential to replace waveform-based techniques at extremely low bit rates - has been presented. Possible extensions to this research are discussed later in this chapter.

7.2 Summary of practical achievements

The extensive review of recent work presented in Chapter 2 led to the conclusion that although there are many problems facing the semantic based video coding technique, it seems to be the only approach potentially capable of compressing the video signal to data rates below 10 kbit/s and retaining an acceptable quality of reconstruction.

As was shown in Chapter 3, the two most difficult issues concerning semantic model based coding are automatic tracking and automatic fitting of the scene model. Although these issues could be considered together, a temporary separation of the two allows more detailed analysis. Such a separation also meant that techniques utilising various mathematical models could be used in each module. This allowed more straightforward testing of new ‘modules’ by simple code replacement. The research concerning wire-frame fitting and tracking was
therefore presented in two separate chapters. In this thesis tracking was presented prior to fitting. This is due to the fact that the experience gathered during the development of the automatic tracking method allowed creation of the most successful fitting algorithm, i.e. historically, the tracking algorithm was created before the most successful fitting algorithm was developed.

A method of automatic tracking of the subject in head-and-shoulders scenes was developed, described and tested in Chapter 4. The method was based on the analysis of principal components of subsequent frames of the video sequence. This approach was facilitated by the high similarity between the adjacent frames. The initialisation of tracking (creation of initial set of facial features) was, at this stage, carried out manually. Detailed guidelines of the extraction method for the important facial features (the eyes, the lips and the nose) were developed and presented.

Several variations of the tracking algorithm were created and tested. These included versions with static and dynamically updated initial sets. For each version two distance measures were applied. One of them described the distance between two individual images (measure $A$), while the other described the distance between an image and a group of images (measure $B$). Since the tracking method was based on PCA, the reduction of the principal component space dimensionality was also taken into account. The tests carried out on widely used video sequences proved that the use of a dynamically updated initial set along with distance measure $A$ provides rather poor results with tracking breaking down relatively quickly. Replacement of measure $A$ with measure $B$ improved the overall robustness, but the tracking of certain facial features broke down on several occasions.

On the other hand, the use of a static initial set provided very reliable results. For both variations of the static initial set algorithm (i.e. utilising measure $A$ and measure $B$) the tracking was maintained in all frames of the tested sequences. However, subjective comparisons seemed to suggest that the distance measure $B$ allowed more accurate tracking of the very centre of the facial features. In order to verify this view, the 2D locations of the important facial features were extracted manually from all frames of the tested sequence. The tracking error was measured as the absolute of the Euclidean distance between the 2D co-ordinates of the features tracked automatically (using measures $A$ and $B$) and manually on a frame-by-frame basis. Analysis of the mean error and standard deviation proved that method $B$ indeed gave better results. For this reason, only a static initial set and measure $B$ were used
in further research. Further tests carried out on a variety of widely used head-and-shoulders sequences confirmed the reliability of the algorithm. The tracking of all the facial features in all tested sequences was maintained. In order to provide a numerical measure of the tracking fidelity, the 2D co-ordinates were extracted from every fifth frame of each tested sequence manually. Error profiles depicting the Euclidean distance between the 2D co-ordinates of the features tracked automatically and manually (every fifth frame) allowed an insight into tracking stability. In the vast majority of cases the mean tracking error did not exceed one pixel, which demonstrated the performance of the algorithm. In the final part of Chapter 4, the suitability of the algorithm for tracking of the vertices describing local motion of a particular facial feature (e.g. eyes' close-open, lips' close-open), was tested. Here also the tracking was maintained.

A solution to the problem of automatic fitting was presented in Chapter 5. Successful fitting of the wire-frame requires accurate localisation of the facial features. Several methods of feature extraction based on purely image processing techniques were proposed and tested. A novel heuristic method of extraction of the centre of the lips was presented. This method utilises the fact that the local motion fields of moving lips have opposite directions. This allows creation of an inter-frame edge detector comparing the positions of pixels describing the upper and the lower lip. The method performed well in situations where the global motion in the scene (i.e. motion of the speaker's body) was negligible. However, it was not possible to detect the centre of the lips reliably in the presence of significant global motion.

In the search for a reliable facial feature extraction algorithm, a method utilising split-and-merge and mathematical morphology techniques was proposed and tested on various head-and-shoulders subjects. Several variations of the algorithm were investigated. For both split and merge stages three separate homogeneity criteria were considered (the dynamic criterion, the variance criterion and the average criterion). The pre-processing of the images using operations of morphological erosion, dilation, opening and closing was also tested. Although the application of mathematical morphology pre-processing along with variance criterion-based split-and-merge technique produced promising result on certain images, this could not be repeated for a wider range of tested subjects. In addition, this technique proved to be very sensitive to the threshold values of split criteria and merge criteria.

At that stage it seemed necessary to broaden the analysis of the problem of automatic wire-frame fitting by adapting both image processing and machine vision techniques. Two
methods utilising a data-base of facial images were developed and tested. In both of them, the same facial data-bases were used.

First, a correlation-based method was tested. It consisted of a preprational (off-line) step and an on-line step. The preprational step involved pre-processing of the facial data-base by application of isotropic edge detection and thresholding in the middle of the greyscale range. Also, each image in the data-base was fitted manually with the Candide scene model. Although the human factor was present in the preprational step, it was absent from the on-line processing. This processing consisted of a two-stage coarse-to-accurate fitting of the wire-frame. In order to judge the accuracy of the automatic wire-frame fitting, animated movies depicting faces fitted manually and automatically were created. An improved algorithm was considered in order to increase accuracy of fitting in the area occupied by lips. Although the overall fitting fidelity increased, so did the computational costs.

The second automatic wire-frame fitting algorithm was based on the analysis of principal components of a facial data-base. Experience gained during the construction of the tracking algorithm (Chapter 4) and the fitting algorithm (earlier sections of Chapter 5) allowed the construction of an entirely new wire-frame fitting algorithm which proved to be faster, more robust and more accurate than the correlation based method. The off-line step involved wire-frame fitting, extraction of sub-sequences containing important facial features (the eyes, the lips and the nose) and entire faces. This resulted in the creation of separate principal component spaces for the faces and the facial features. Here also the fitting results were assessed subjectively after the creation of short animated movies. Although this approach was painstaking and lengthy, it provided far more meaningful results than waveform-based error measures (e.g. PSNR). Since in real world situations it is human vision which is the reference point for any evaluation of accuracy/fidelity of image or video processing.

The facial features' tracking algorithm presented in Chapter 4 was initialised manually. The results of the research presented in Chapter 5 provided a tool for automatic fitting of a model to the actual scene. A unified approach to the two most important problems in semantic model based video coding: automatic model fitting and automatic subject tracking, was therefore the subject of Chapter 6.

However, a simple fusion of the algorithms presented in Chapter 4 and 5 was not possible. Some pre-processing, both in the off- and on-line stages proved necessary. This additional
pre-processing consisted of spatial and histogram equalisation. The new fitting/tracking algorithm consisted of an initial tracking stage and a standard tracking stage. In the initial tracking stage, localisation of the important facial features took place without human intervention. These features were localised using principal component analysis of a generic histogram-equalised reference data-bases containing faces and facial features.

As a result of tests carried out on a wide range of head-and-shoulders scenes it was found, that the newly developed algorithm was capable of tracking all the important facial features in all tested sequences. The statistical analysis showed, that the replacement of manual initialisation by automatic initialisation gave slightly poorer results (in terms of mean error and standard deviation). However, there was no visible difference in the tracking performance between the methods presented in Chapter 6 and Chapter 4 apart from slightly different tracking paths being visible on a very few frames.

As neither off-line manual data-base preparation, nor evaluation of the results, were parts of the on-line algorithm, it is fair to say, that the human factor has been entirely removed from this work. This is a valuable achievement, since as can be seen in the proceedings of the MPEG-4 group, although the coding of particular segments of the image is well defined, a way to segment the scene into meaningful objects is still not fully established.

As discussed in summary of Chapter 6, since the overall speed performance was quite satisfactory - even using modest processing platform - only limited attention was paid to computational efficiency.

7.3 Future work

The methods proposed in this thesis constitute a solid basis for further development. Although reliable tracking was achieved, the described algorithms do not utilise explicit knowledge about the structure of the human face. It is only in the accurate stage of fitting, that the fact that the eyes and the lips occupy certain areas in the human face is utilised. Compared to other approaches, which utilise heuristics (detailed proportions of the face, parametrisation of the face, etc.) it is fair to say, that the semantics in the presented approach are used only when necessary, and even then with a very reduced set of rules.
Both fitting and tracking algorithms follow a relatively regular scheme. Also, both techniques are based on the same mathematical methods. This would simplify hardware implementation. Since the algorithm is virtually the same for all the facial features, a parallel machine adaptation would be rather a trivial task. However, it is almost certain, that a very low-cost real-time hardware application would be a better and indeed cheaper option. A software only solution might also be possible given suitable hardware.

As has been mentioned before, the proposed algorithm almost never utilises the knowledge about the geometry of the human face. Should this prove necessary, the overall robustness of the algorithm could be increased by the introduction of additional 'fail-safes' in the form of a semantic based tracking control (Figure 7.1). Should tracking of one facial feature fail, it could always be recovered, or maintained, by taking into account the positions of the remaining - properly tracked - facial features. As has been established throughout this research, tracking of certain facial features proves to be very reliable. This is especially true of the eyes. The unique combinations of bright and dark pixels allows recovery of the position of the left and the right eye (whenever the resolution of the sequence allows) with great accuracy. Although the above is not necessary at the present moment, it could be taken into account when analysing more complicated scenes, than head and shoulders.

Figure 7.1: Semantic based tracking control

Continuation of this research in the area of fitting and tracking should target construction of a device capable of tracking and transmitting the position of the vertices describing the facial features in real time. Although the author felt that a real-time application was not crucial at the research stage, application of the method in a hardware device would be a logical follow up. Before this is done however, the issues concerning lip synchronisation and appropriate
model selection (i.e. when the scene contains more than one speaker - e.g. teleconferencing scenario) must also be addressed.

Another possibility is a combination of the presented method with one of the waveform-based techniques in form of a layered coder. Probably the most promising techniques in the area of waveform based coding are the ones utilising wavelet analysis. In this case the method described in this thesis would serve as a tool for localisation of important facial features in the scene. This would allow a semantic based change in the quantisation step of the waveform based codec, so that coding quality could be increased in the area occupied by the face and especially for important facial features. As can be seen in Figure 7.2, elimination of artefacts in the area of important facial features (particularly the lips) greatly increases the subjective quality of an image in a videophone scene.

Figure 7.2: Model-waveform hybrid codec possible results
7.4 Model based approach to moving image coding: final remarks

Research in the field of extremely low data-rate moving image coding, after a time of relative stalemate, seems to be gaining momentum. This is mainly due to the rapid development of mobile communications and other new technologies operating in a relatively low bandwidth (e.g. Internet). This view is supported by the work of the MPEG-4 group. The renewed interest in model based coding techniques is also visible in the number of papers recently published in the Transactions of the IEEE concerning video coding. In a special issue devoted to very low bit rate video compression [ieee94] there was not a single article concerning DCT block based compression techniques and the importance of both object and knowledge based techniques was acknowledged. While there were three articles describing each model based approach, there were only two articles concerning subband/wavelet coding and one article on fractals. Although the DCT block based techniques have gained ground recently (mainly due to creation of the H.263 standard), it is fair to say that, model based techniques constitute at the present moment the mainstream of very low bit-rate research, certainly within the scope of the MPEG-4 group. In a recent special issue of IEEE Transactions on Circuits and Systems for Video Technology [ieee97], the greatest number of articles concern techniques more or less targeting a model-based approach. These are followed by wavelet based, and block DCT based approaches (mainly utilising the achievements of H.263).

The fact that there is a lower bandwidth limit to waveform based techniques seems to be widely accepted, as is the fact that model based techniques face far greater problems. These will not be resolved this year, or next year or any time in the near future, if there is not enough research focused on the problems of model based coding. This thesis proposes a solution to certain problems concerning a model based approach. The positive feedback the author has received from his numerous published papers, confirms that a solution to the main problems facing semantic based coding (automatic fitting and automatic tracking of the scene model) is very much needed and looked for, and that the fitting and tracking methods described in this thesis constitute a valuable contribution to this area of research.
References


[bedford94] T. Bedford, F. M. Dekking, M. Breeuwer, M. S. Kane and D van Schooneveld, 'Fractal coding of monochrome images', Signal


References

Workshop on Coding Techniques for Very Low Bit-Rate Video, Colchester, UK, April 1994, Paper number: 3.4.


[eleftheriadis95-2] A. Eleftheriadis and A. Jacquin, ‘Automatic face location detection and


References


References

Analysis and Machine Intelligence, vol. 15, no. 6, June 1993, pp. 545-555.


References


[peipa] http://peipa.essex.ac.uk/


References


References


S. Yao and R.J. Clarke, ‘Motion-compensated wavelet coding using adaptive vector quantisation’, IEE Colloquium on Applications of


Appendix 1

Code example: histogram equalisation

The code for the presented research was written in C. However, certain elements (especially concerning handling of facial data-base of images) were written in C++. The final version of software consists of 16 source files, about 250 lines of code each. Since all the algorithms but histogram equalisation were described in detail in the main part of the thesis, here only the histogram equalisation code is presented as an example. Presentation of the entire software code would expand this thesis to unacceptable proportion. The following sample (listing from one file only) gives an idea about the size of entire package.

The histogram equalisation algorithm was written in the way, so that it would be necessary to use externally only one function:

```c
void ModifyHistogram(char** Image, long DXIma, long DYIma, long Levels, double* OrdHist)
```

taking five arguments: a 2D table containing greyscale image (`Image`: one byte per pixel), its dimensions: `DXIma` and `DYIma`, and number of greyscales `Levels`). The `OrdHist` variable contains the table with an ordered histogram. If the histogram is to be linearised (equalised), this parameter is set to `NULL`.

Another function that can be used externally is `PlotHistogram`. It allows to visualise the histogram and examine the effectiveness of the algorithm. `PlotHistogram` uses function `SetPGM` not described here. `SetPGM` (defined in another file, here declared in `util.h` header) saves any 2D table in form of a portable greyscale bitmap file. `GetImageInfo` function used in the application example is also defined in a separate file. It determines the type of analysed image file (i.e. hex, PGM, BMP or other) and retrieves the dimensions of the bitmap image.

Histogram $h(l)$ (where $l$ denotes greylevel) is the probability density of greyscales in the image (Figure A.1). We are looking for certain function $s=T(l)$ (where $s$ denotes a greylevel
in the new co-ordinate system) that would transform $h(l)$ into a new histogram: $h'(s)$ (preferably linearised - as shown in Figure A.1).

The $h(l)$, $T(l)$ and $h'(s)$ functions are usually normalised, so that they fit into the [0, 1] range. In the discrete case, this range corresponds all greylevels in the image ([0, 255] in the case of the presented software application example).

The code follows elementary probability theory which says that the probability density of the transformed histogram can be described by the following relationship (a.1):

$$ h'(s) = h(l) \frac{dl}{ds} \quad (a.1) $$

Let us consider the following transformation function $T(l)$:

$$ s = T(l) = \int h(l)dl \quad (a.2) $$

If derivative after $l$ is calculated for both sides of (a.2):

$$ \frac{ds}{dl} = h(l) \quad (a.3) $$

Considering relationships (a.1) and (a.3):

$$ h'(s) = h(l) \frac{1}{h(l)} = 1 \quad (a.4) $$
which says, that in order to obtain linear histogram $h(s)$, the transformation function must be
integral of the original histogram (a.2). The integral of the histogram is often referred to as
'cumulative histogram'. The cumulative histogram is created by the function Cumulate in the
example code.

```c
#include <stdio.h>
#include <stdlib.h>
#include "const.h"
#include "util.h"

/* calculate the histogram of the Image of dimensions DXIma, DYIma */
/* Levels greyscales and put it into the Hist table, a matrix of */
/* DXIma x DYIma char elements and a table of Level double elements */
/* must be allocated beforehand */

void Histogram(char** Image, double* Hist, long DXIma, long DYIma, long Levels)
{
    long r, c;
    long ImaSize = DXIma * DYIma;
    for(r = 0; r < Levels; r++)
        Hist[r] = 0.0;
    for(r = 0; r < DYIma; r++)
        for(c = 0; c < DXIma; c++)
            Hist[ (unsigned char)Image[r][c] ] += 1.0;
    for(r = 0; r < Levels; r++)
        Hist[r] /= (double)ImaSize;

    /* create a cumulative histogram of Hist and put it back to Hist table, */
    /* table of Levels double elements must be allocated, the contents */
    /* of original histogram is destroyed */
```
void Cumulate(double* Hist, long Levels)
{
    long r;

    for(r = 0; r < Levels - 1; r++)
        Hist[r + 1] += Hist[r];
}

/* create lookup table for equalised histogram using cumulative */
/* table of original histogram and put it into LookUp, all tables */
/* must be allocated beforehand */

void CreateEqualisingLookUp(double* CumHist, unsigned char* LookUp, long Levels)
{
    long r;
    double Approx, Diff;
    unsigned char Trunc;

    for(r = 0; r < Levels; r++)
    {
        Approx = CumHist[r] * (Levels - 1);
        Trunc = (unsigned char)Approx;
        Diff = Approx - (double)Trunc;

        if(Diff < 0.5)
            LookUp[r] = Trunc;
        else
            LookUp[r] = Trunc + 1;
    }
}

/* create ordered lookup table for ordered histogram using */
/* lookup table of the equalised image and cumulative version */
/* of the required histogram, the LookUp variable initially */
/* contains the equalising lookup table, after the function */
/* is executed, it is replaced by ordered lookup table */

void CreateOrderedLookUp(unsigned char* LookUp, double* OrdCumHist, long Levels)
{
    double MinValue, Buffer, OrdLookEl;
    long MinIndex, r, c;

Appendix 1: Code example: histogram equalisation

for(c = 0; c < Levels; c++)
{
    OrdLookEl = (double)LookUp[c] / (Levels - 1);
    MinValue = OrdCumHist[0] - OrdLookEl;
    if(MinValue < 0.0)
        MinValue = -MinValue;
    MinIndex = 0;

    for(r = 1; r < Levels; r++)
    {
        Buffer = OrdCumHist[r] - OrdLookEl;
        if(Buffer < 0.0)
            Buffer = -Buffer;

        if(Buffer < MinValue)
        {
            MinValue = Buffer;
            MinIndex = r;
        }
    }

    LookUp[c] = MinIndex;
}

/* convert the Image using the LookUp table */

void Convert(char** Image, unsigned char* LookUp, long DXIma, long DYIma)
{
    long r, c;

    for(r = 0; r < DYIma; r++)
        for(c = 0; c < DXIma; c++)
            Image[r][c] = (unsigned char)LookUp[(unsigned char)Image[r][c]];
}

/* draw histogram to a pgm file */

void PlotHistogram(double* Hist, long Levels, long MaxPix, char* Name)
{
    long r, c, YPix;

double Max, Min, Range;
char** Plot;

Max = Hist[0];
Min = Hist[0];

for(r = 1; r < Levels; r++)
{
    if(Hist[r] < Min)
        Min = Hist[r];
    if(Hist[r] > Max)
        Max = Hist[r];
}

Range = Max - Min;
if(Range == 0)
    Range = 1/(double)Levels;

Plot = charmatrix(0, Levels - 1, 0, MaxPix - 1);
for(r = 0; r < Levels; r++)
    for(c = 0; c < Levels; c++)
        Plot[r][c] = 255;

for(r = 0; r < Levels; r++)
{
    YPix = (long) ((Hist[r] * (double)MaxPix) / Range);
    YPix--;
    if(YPix >= MaxPix)
        YPix = MaxPix - 1;
    if(YPix < 0)
        YPix = 0;
    DrawVLine(Plot, r, MaxPix - 1 - YPix, MaxPix - 1, 0);
}

SetImage(Name, Plot, Levels, MaxPix, PGM);
free_charmatrix(Plot, 0, Levels - 1, 0, MaxPix - 1);

/* modify histogram of an image, Levels tells the number of greyscales */
/* if the OrdHist parameters is NULL, the histogram of the image is */
/* equalised, otherwise, the histogram is approximated to that */
/* requested in the OrdHist table */

void ModifyHistogram(char** Image, long DXIma, long DYIma, long Levels, double* OrdHist)
{
    double *Hist;
unsigned char* LookUp;

Hist = (double*)malloc(Levels * sizeof(double));
LookUp = (unsigned char*)malloc(Levels * sizeof(unsigned char));

Histogram(Image, Hist, DXIma, DYIma, Levels);
Cumulate(Hist, Levels);
CreateEqualisingLookUp(Hist, LookUp, Levels);

if(OrdHist != NULL)
{
    Cumulate(OrdHist, Levels);
    CreateOrderedLookUp(LookUp, OrdHist, Levels);
}

Convert(Image, LookUp, DXIma, DYIma);

free((char*)Hist);
free((char*)LookUp);

/* application example */

void main(int argc, char** argv)
{
    long DXIma, DYIma, Levels, Format;
    char** Image;
    double *Hist, *OrdHist;
    unsigned char* LookUp;

    printf("- histogram manipulation\n");
    if(argc < 5)
    {
        printf("Usage: %s image DX DY levels\n", argv[0]);
        return;
    }

    Format = GetImageInfo(argv[1], &DXIma, &DYIma);
    Levels = 256;

    if(Format == NIL)
    {
        Format = HEX;
        scanf(argv[2], "%i", &DXIma);
        scanf(argv[3], "%i", &DYIma);
        scanf(argv[4], "%i", &Levels);
    }

    Image = charmatrix(0, DXIma - 1, 0, DYIma - 1);
Appendix 1: Code example: histogram equalisation

Hist = (double*)malloc(Levels * sizeof(double));
OrdHist = (double*)malloc(Levels * sizeof(double));
LookUp = (unsigned char*)malloc(Levels * sizeof(unsigned char));

GetImage(argv[1], Image, DXima, DYima, Format);

Histogram(Image, Hist, DXima, DYima, Levels);
PlotHistogram(Hist, Levels, 256, "historg");

ModifyHistogram(Image, DXima, DYima, Levels, NULL);

Histogram(Image, Hist, DXima, DYima, Levels);
PlotHistogram(Hist, Levels, 256, "histequ");

setImage(argv[1], Image, DXima, DYima, PGM);

free_charmatrix(Image, 0, DXima - 1, 0, DYima - 1);
free((char*)Hist);
free((char*)OrdHist);
free((char*)LookUp);
}
Published articles

The research presented in this PhD thesis has been published in the following journals and conferences.

Journal publications:


Conference publications:


