INFORMATION TO USERS

This manuscript has been reproduced from the microfilm master. UMI films the text directly from the original or copy submitted. Thus, some thesis and dissertation copies are in typewriter face, while others may be from any type of computer printer.

The quality of this reproduction is dependent upon the quality of the copy submitted. Broken or indistinct print, colored or poor quality illustrations and photographs, print bleedthrough, substandard margins, and improper alignment can adversely affect reproduction.

In the unlikely event that the author did not send UMI a complete manuscript and there are missing pages, these will be noted. Also, if unauthorized copyright material had to be removed, a note will indicate the deletion.

Oversize materials (e.g., maps, drawings, charts) are reproduced by sectioning the original, beginning at the upper left-hand corner and continuing from left to right in equal sections with small overlaps.

Photographs included in the original manuscript have been reproduced xerographically in this copy. Higher quality 6" x 9" black and white photographic prints are available for any photographs or illustrations appearing in this copy for an additional charge. Contact UMI directly to order.

Bell & Howell Information and Learning
300 North Zeeb Road, Ann Arbor, MI 48106-1346 USA

UMI®
800-521-0600
Spatial and Temporal Prediction Schemes for Object-Based Digital Video

by

Nicos Herodotou

A thesis submitted in conformity with the requirements for the degree of Doctor of Philosophy
Department of Electrical and Computer Engineering
University of Toronto

©Copyright by Nicos Herodotou 1998
The author has granted a non-exclusive licence allowing the National Library of Canada to reproduce, loan, distribute or sell copies of this thesis in microform, paper or electronic formats.

The author retains ownership of the copyright in this thesis. Neither the thesis nor substantial extracts from it may be printed or otherwise reproduced without the author’s permission.

0-612-41438-8
Acknowledgements

I wish to express my sincere gratitude to my thesis supervisor Prof. A.N. Venetsanopoulos, not only for his academic guidance but also for sharing his knowledge of life. My thanks also go to my colleagues, and also to Dr. Kostas Plataniotis for his suggestions towards this work. I would also like to thank the committee members and Prof. Katsaggelos for their useful comments.

Finally, I wish to express my greatest appreciation for my parents Phidias and Olvia Herodotou, and my wife Lilian for being there through many difficult times.
Abstract

Object-based video compression schemes have recently gained tremendous interest, and are the driving force of the future MPEG 4 and 7 video coding standards. These object-based coders not only provide for a higher compression efficiency, but they can also support content-based functionalities for future multimedia applications. In this thesis, spatial and temporal prediction techniques are investigated for these newer object-based compression schemes. Structural and statistical models are first investigated for spatially interpolating digital images for editing/compositing purposes in object-based video or for frame-based compression systems. Nonlinear filter structures based on vector order statistics are selected for spatially interpolating color images due to their robustness, preservation of edge information and image details, and their ability to exploit the interchannel color correlations. Statistical methods based on Gibbs random field (GRF) models are also used to obtain an interpolated
image. The iterative GRF methods are approximated by a non-iterative nonlinear filtering operation, thereby reducing the computational complexity of the process. The proposed methods are all compared to the conventional linear approaches. Temporal prediction of videophone-type sequences is subsequently investigated in view of the problems associated with conventional block-based methods. Segmentation-based motion compensated prediction is carried out by partitioning an image into a facial region and a set of arbitrarily-shaped regions. A novel and useful approach is presented to automatically locate and track the facial area within image sequences using the visual cues of color and shape. The scheme is robust in scenes with head-and-shoulders type images found in videophone sequences or facial image databases, and can be easily tuned for more general cases where many faces may be present in the scene. A general color segmentation technique is then proposed to partition the remainder of the image and form a triangular mesh model. This is subsequently followed by a suitable mesh tracking scheme. Motion compensated prediction is finally performed by utilizing the estimated nodal point motion vectors and an affine warping transformation. Experimental results provide a comparison between the proposed scheme and the conventional block-based methods.
# Contents

1 Introduction .......................................................... 1
   1.1 Thesis Objectives and Outline .................................. 5

2 Spatial Image Interpolation Using Nonlinear Filter Structures 10
   2.1 Introduction .......................................................... 10
   2.2 Digital Resampling .................................................. 12
       2.2.1 Sampling Rate Reduction by an Integer Factor .......... 13
       2.2.2 Increasing the Sampling Rate by an Integer Factor ..... 14
   2.3 Conventional Interpolation Methods ............................ 17
   2.4 Order Statistic Filters .............................................. 19
       2.4.1 Median Hybrid Filters ........................................ 21
   2.5 Multichannel Data .................................................. 22
       2.5.1 Ordering of Multivariate Data .............................. 23
   2.6 Vector Median Type Filters ....................................... 24
       2.6.1 Rectangular and Quincunx Resampled Lattices .......... 25
   2.7 Image Interpolation Using Vector-FIR Median Hybrid Filters 27
   2.8 Experimental Results ............................................. 30
   2.9 Conclusions ......................................................... 33

3 Statistical Methods for Spatial Image Interpolation 50
   3.1 Introduction .......................................................... 50
   3.2 Modeling Images using Gibbs Random Fields .................... 51
   3.3 Parameter Estimation ................................................. 54
3.4 Image Interpolation .......................................................... 56
   3.4.1 Nonlinear Filtering Using Binary Logic-Type Operations ... 60
3.5 Experimental Results ....................................................... 61
3.6 Conclusions ................................................................. 63

4 Facial Image Segmentation .................................................. 66
4.1 Motivation and Related Work ............................................ 66
4.2 Color Image Segmentation ................................................. 68
   4.2.1 Color Coordinate Systems ........................................... 68
   4.2.2 Color Space Selection ............................................... 72
   4.2.3 Skin-tone Distribution .............................................. 75
   4.2.4 Extraction of Skin-tone Regions .................................... 77
   4.2.5 Segmentation Using the Color Attribute ........................... 82
   4.2.6 Median Filtering and Region Filling/Removal ....................... 84
   4.2.7 Morphological Processing ............................................ 85
4.3 Shape and Color Analysis ............................................... 87
   4.3.1 Introduction .......................................................... 87
   4.3.2 Fuzzy Membership Functions ....................................... 90
   4.3.3 Aggregation Operators ............................................... 93
   4.3.4 Metadata and Content-Based Retrieval ............................. 98
4.4 Experimental Results ...................................................... 99
   4.4.1 Akiyo Sequence ....................................................... 102
   4.4.2 Miss America Sequence ............................................. 105
   4.4.3 Foreman Sequence .................................................... 106
4.5 Conclusions ................................................................. 107

5 Motion Compensated Prediction ............................................ 130
5.1 Introduction ................................................................. 130
5.2 Intraframe Compression Methods ....................................... 133
   5.2.1 Lossless Compression ............................................... 134
   5.2.2 Lossy Compression Techniques ...................................... 135
5.2.3 Second Generation Coding Methods .................................. 137
5.3 Interframe Compression Techniques ................................. 138
  5.3.1 3-D Waveform Coding ........................................ 138
  5.3.2 Motion Compensated Waveform Coding .................. 139
  5.3.3 Motion Estimation .......................................... 140
    5.3.3.1 Optical Flow-based Techniques .................. 141
    5.3.3.2 Pel-recursive Methods ............................. 143
    5.3.3.3 Block Matching Methods ......................... 143
    5.3.3.4 Deformable Block Matching .................. 145
  5.3.4 Analysis-Synthesis Coding ................................ 147
5.4 Object-based Coding ............................................. 149
  5.4.1 Mesh-based Modeling ..................................... 150
  5.4.2 Color Image Segmentation ................................ 154
    5.4.2.1 Histogram Thresholding ......................... 156
    5.4.2.2 Post-Processing and Region Merging ........ 157
  5.4.3 Content-Based Mesh Design ................................ 158
  5.4.4 Mesh Tracking ........................................... 164
  5.4.5 Image Warping Prediction ................................ 169
5.5 Experimental Results ............................................ 173
  5.5.1 Color Image Segmentation ................................ 173
  5.5.2 Mesh Design .............................................. 175
  5.5.3 Mesh Tracking ............................................ 175
  5.5.4 Image Warping Prediction ................................ 176
5.6 Conclusions ..................................................... 182

6 Conclusions and Future Directions ............................... 193

Bibliography .......................................................... 198
List of Figures

2.1 General system for sampling rate reduction by M. .................................................. 14
2.2 General system for sampling rate increase by L. ......................................................... 14
2.3 (a) Rectangular decimation (b) Quincunx decimation. ............................................... 26
2.4 Applications of image/video zooming: (a) image zoom and transfer of circular object from video sequence “A” to video sequence “B” by editing and compositing as envisioned in MPEG-4 and 7, and (b) a video sequence is transmitted at a reduced spatial resolution and scaled up to an increased resolution for playback at the receiver. ................................................................. 36
2.5 Frequency-domain illustration of downsampling [16]. ............................................... 37
2.6 Frequency-domain illustration of interpolation [16]. .................................................. 38
2.7 The magnitude of the Fourier Transform provides a comparison of the different interpolating functions in the frequency domain. The dashed box indicates the ideal lowpass filter response: (a) Nearest neighbor, (b) Bilinear, (c) Cubic B-spline, and (d) Cubic spline. ................................................................. 39
2.8 Spatial image interpolation of Lena image: (a) Original image of Lena, (b) downsampled image of the original (i.e. by a factor of 4 in each direction without prefiltering), (c) Pixel replication, (d) Bilinear technique, (e) Cubic spline, (f) FMH_{R6}, (g) VMF, (h) VFMH_{R2}, (i) VFMH_{R4}, (j) VFMH_{R6}. .................................................. 40
2.9 Spatial image interpolation of a synthetic image: (a) Original synthetic image, (b) downsampled image of the original (i.e. by a factor of 4 in each direction without prefiltering), (c) Pixel replication, (d) Bilinear technique, (e) Cubic spline, (f) FMH_{R6}, (g) VMF, (h) VFMH_{R2}, (i) VFMH_{R4}, (j) VFMH_{R6}.

3.1 Hierarchically arranged neighborhood systems $\eta^m$ [46]...

3.2 Neighborhood systems $\eta^1$ and $\eta^2$ and their associated clique types [46].................................

3.3 (a) Original Mandrill image, (b) a textured region of the original image, (c) pixel replication, (d), bilinear interpolation, (e) Gibbs Sampler, (f) Simulated Annealing, (g) Iterative Conditional Mode, and (h) the 3-pass nonlinear filter.

3.4 (a) Original Lena image, (b) a smaller section of the original image, (c) pixel replication, (d), bilinear interpolation, (e) Gibbs Sampler, (f) Simulated Annealing, (g) Iterative Conditional Mode, and (h) the 3-pass nonlinear filter.

4.1 HSV hexcone color model.

4.2 Skin and Lip Clusters in the RGB color space model.

4.3 Skin and Lip Clusters in the L*a*b* color space model.

4.4 Skin and Lip Hue Distributions in the HSV color space model.

4.5 Skin color distributions of different races in the HSV space, a) Caucasian, b) African-American, and c) Asian test samples.

4.6 HSV hexcone model at different values of the V component, a) $V=100\%$, b) $V=63\%$, c) $V=T_{Val}=35\%$, and d) $V=20\%$.

4.7 Overall Segmentation scheme using Color and Shape attributes.

4.8 Hue values of the Principal Polyhedron along with its Scale-Space filtered version for Frame 20 of the following video sequences: a) Claire, b) Miss America, c) Akiyo, d) Foreman, e) African-American sample image, and f) Carphone.
4.9 Effect of the SP and the Saturation threshold of the PP on the segmentation process: a) Original Frame 20 of the Carphone sequence, b) results using the PP and SP Stages in the segmentation process as proposed in Figure 4.7, c) output obtained when using only the PP Stage of the extraction process, and d) the results found when the Saturation threshold in the PP is reduced to a value of 15%.

4.10 Extraction of the facial region for the Akiyo sequence (Frame 20) through the various stages, a) Initial extraction by the Principal Polyhedron for the Hue Region, $8^\circ \leq HR_1 \leq 50^\circ$ b) Median filtered result, c) Region filling and removal, d) Extraction of the Secondary Polyhedron for HR$_1$, e) Region merging of the extracted PP and SP regions, f) Final result of HR$_1$ after morphological processing (i.e. C$_1$), g) Shape processing of object, C$_1$ (i.e. facial region), and h) Shape processing of object, C$_2$ found from the second Hue Region $-20^\circ \leq HR_1 \leq 8^\circ$.

4.11 Saturation components of the Principal Polyhedron for the two Hue Regions, HR$_1$ and HR$_2$, of the Akiyo sequence (Frame 20).

4.12 Extraction of the facial region for the Miss America sequence (Frame 20), a) Extraction of the facial region prior to morphological processing, b) Shape processing of object, C$_1$ (i.e. facial region) in Hue Region, HR$_1$, c) Shape processing of object, C$_2$ in Hue Region, HR$_2$, and d) Shape processing of object, C$_3$ in HR$_2$.

4.13 Extracted objects to be analyzed by the shape and color analysis module: a) Original Frame 20 of the Foreman sequence, b) the object in Hue Region, HR$_1$ (i.e. facial region), c) the object in Hue Region, HR$_2$, and d) the two candidate objects in Hue Region, HR$_3$. 

4.14 The effects of glasses on the facial extraction process:  
a) Original Image #1 from the M2VTS database,  
b) initial extraction by the Principal Polyhedron,  
c) final segmented image.

4.15 Location and tracking of the facial region for the following images and video sequences:  
a) Claire,  
b) Miss America,  
c) Akiyo,  
d) Foreman,  
e) African-American sample image,  
f) Carphone,  
g) M2VTS Image #1,  
h) M2VTS Image #2.

5.1 The source encoding process.

5.2 Block matching in backward motion estimation.

5.3 Deformable block matching.

5.4 Motion Compensation with spatial transformations using a rectangular mesh structure.

5.5 A content-based 2D mesh design for motion compensation.

5.6 Nodal point selection and triangulation for a segmented region when all boundary points connect to the center of mass, C1.

5.7 Nodal point selection and triangulation for a segmented region when several boundary points cannot connect to the center of mass, C1. In this case the region R1 is split into two sub-regions, R11, comprised of (B1, B2, B3, B4, B5, B6, B10, B11, B12, B13, C1), and R12 consisting of (B6, B7, B8, B9, B10, C2).

5.8 Nodal point selection and triangulation for a segmented region when the center of mass, C1 falls outside the region of interest, R1. In this case, R1 is split vertically into two sub-regions, R11, and R12 at the horizontal midpoint of the bounding rectangle.
5.9 Nodal placement and triangulation scheme for the general case where a scene is partitioned into arbitrary regions: (a) segmented image consisting of four disjoint regions, $R_i$, for $i = 1, \ldots, 4$, and (b) the resulting mesh generation. 162

5.10 Block structure and weighting factors proposed in the modified block matching technique. 165

5.11 Modification of the point matching process for nodal points in the vicinity of the facial region: (a) the facial region, $R_f$ in frame $k$ is shown in the foreground of two stationary regions, $R_1$, and $R_2$, and is about to move horizontally in the subsequent frame, $(k+1)$, (b) the normal point matching process leads to slight distortions in the predicted (reconstructed) frame $(k+1)$, and (c) the nodal points $p_4$, and $p_5$ which are covered by the facial region, remain stationary and the facial area is overlaid over the reconstructed image eliminating the distortion. 167

5.12 (a) Two regions, $R_1$, and $R_2$ which undergo complex motion within the scene, and (b) region, $R_3$ is uncovered as a result of the motion of nodal point, $p_1$ from the previous frame, $k$ to position, $p_1'$ in the predicted frame, $k+1$. 169

5.13 General structure of a predictive encoder/decoder in a video compression scheme. 172

5.14 Segmentation of the Carphone sequence: (a) Original Frame 80, (b) extraction of the facial region, (c) the chromatic regions, (d) the achromatic regions, (e) the transitional chromatic/achromatic regions, (f) segmentation before region merging, and (g) final segmentation after region merging. 184

5.15 Segmentation of the Claire sequence: a) Original Frame 100 of the Claire sequence, b) Extraction of the facial region, c) Final Segmentation after Region Merging. 185
5.16 (a) Nodal point selection along the boundaries of the segmented regions in Frame 80 of the *Carphone* sequence, and (b) mesh generation. ............................................................... 186

5.17 (a) Nodal point selection along the boundaries of the segmented regions in Frame 100 of the *Claire* sequence, and (b) mesh generation. ............................................................... 186

5.18 Nodal point selection along the boundaries of the segmented regions in Frame 80 of the *Carphone* sequence, (b) tracking of nodal points in Frame 85, and (c) the tracking of nodal points in Frame 90. ............................................................... 187

5.19 (a) Nodal points (i.e. white pixels) covered by the facial area (i.e. black area) of the predicted frame, and (b) pixels on the boundary of the facial area in the previous frame which are not covered by the face in the predicted frame are constrained to lie on the boundary in the predicted frame (i.e. the red outline). ............................................................... 187

5.20 (a) Nodal point selection along the boundaries of the segmented regions in Frame 100 of the *Claire* sequence, and (b) tracking of nodal points in Frame 110. ............................................................... 188

5.21 Motion compensated prediction of the *Carphone* sequence: (a) original Frame 80, (b) original Frame 85, (c) motion compensated prediction of Frame 85 using standard block matching (BMA), and (d) motion compensated prediction of Frame 85 using warping prediction. ............................................................... 189

5.22 An efficient nodal placement strategy that reduces the number of necessary motion vectors. ............................................................... 189
5.23 Motion compensated prediction of the *Carphone* sequence: (a) original Frame 80, (b) original Frame 90, (c) motion compensated prediction of Frame 90 using standard block matching (BMA), and (d) motion compensated prediction of Frame 90 using warping prediction.

5.24 Motion compensated prediction of the *Claire* sequence: (a) original Frame 100, (b) original Frame 110, (c) motion compensated prediction of Frame 110 using standard block matching (BMA), and (d) motion compensated prediction of Frame 110 using warping prediction.
# List of Tables

2.1 A quantitative comparison of the different interpolation methods using rectangular decimation for *Lena* and the synthetic image ........................................ 34

2.2 A quantitative comparison of the different interpolation methods using quincunx decimation for *Lena* and the synthetic image 34

2.3 A quantitative comparison of the nonlinear interpolators with other sample images for the case of rectangular decimation 35

4.1 Statistics of the Hue distribution categorized by race. . . . . 77

4.2 Akiyo (Width × Height=352×288) Shape & Color Analysis. . 104

4.3 Miss America (Width × Height=360×288): Shape & Color Analysis. ......................................................... 105

4.4 Foreman (Width × Height=176×144): Shape & Color Analysis. 106

5.1 Partitioning of the HSV hexcone model. ......................... 155

5.2 Summary of Motion Compensated Prediction for Frame 85 of the *Carphone* sequence. ........................................ 177

5.3 Summary of Motion Compensated Prediction for Frame 90 of the *Carphone* sequence. ........................................ 180

5.4 Summary of Motion Compensated Prediction for Frame 110 of the *Claire* sequence. ................................. 181
Chapter 1

Introduction

An increasing number of multimedia applications are rapidly emerging with the tremendous growth of the Internet, and the recent advances in both, hardware and software development. Digital video is a fundamental media type that is commercially popularizing a wide range of applications such as videoconferencing, mobile videophones, digital video broadcasting, video on demand, and multimedia collaboration products over the Web. The digital representation of video offers numerous advantages over its analog counterpart, including: (i) an open architecture where video may exist at various spatial, temporal, and Quality of Service resolutions, (ii) interactivity, (iii) variable-rate transmission, (iv) easy software conversion amongst standards, (v) editing capabilities such as cut and paste, zooming, and compositing, (vi) robustness to channel noise, and (vii) ease of encryption.

The efficient digital representation of image and video signals is a fundamental component in enabling the applications and technologies mentioned above. The raw data rate required to transmit an HDTV (High Definition TV) color video signal, at a resolution of 1920 pixels × 1080 lines at 8 bits/pixel per color, and 30 frames/sec is 1.5 Gb/s (Gigabits per second). This is indeed an enormous channel capacity requirement considering that conventional modem access over analog telephone lines is currently at 56 kb/s. Similarly, the storage requirements for 75 minutes of uncompressed digital video at an NTSC (National Television Standards Committee) resolution of 720 pixels × 480 lines, at 8 bits/pixel per color, and 30 frames/sec is approximately 140 GB
(GigaBytes). This far exceeds the capacities of single hard disk drives currently in the market where 1 to 30 GB of storage are typical, as well as CD-ROM technologies where conventional CD’s (Compact Disks) are capable of holding 650 MB and the more recent DVD (Digital Versatile Disk) drives at approximately 5 GB. Thus, video compression is essential in achieving the necessary bitrate reductions for both, storage and transmission purposes.

With the tremendous commercial interest in digital video, the need for international image and video compression standards arose. For this purpose, the Moving Picture Experts Group (MPEG) was formed to develop appropriate coding standards. The standardization efforts aimed to serve two important purposes: (i) to ensure interoperability of products from different vendors, and (ii) to facilitate an increased demand where a mass production of VLSI (Very Large Scale Integration) systems and devices would lead to *economies of scale* [1].

MPEG-1 was the first standard that emerged and it was completed in 1992. It encompassed the coded representation of moving pictures and their associated audio at targeted bitrates up to 1.5 Mb/s. It was a successful standard and a de-facto form of storing digital video on the World Wide Web and also used in millions of Video CD’s. MPEG-2 subsequently followed (released in 1994) as an extension to MPEG-1, and was intended for applications that demanded higher qualities. MPEG-2 was given the charter to provide video quality equal to or better than NTSC television with bitrates up to 10 Mb/s. This latter standard has been the timely response for the satellite broadcasting and cable television industries. Millions of set-top boxes have been sold in the last three years using MPEG-2 technology. The efforts of the MPEG body have created a truly successful synergy between industry and academia.

Video sequences, in general, contain significant statistical redundancies in both, the temporal and spatial directions. As a result of this, the magnitude of a particular image pixel can usually be determined from the surrounding pixels within the same frame (using intraframe coding methods) or from pixels of a nearby frame (i.e. interframe techniques). The intraframe compression algorithms in the MPEG standards utilize the DCT (Discrete Cosine Transform) on image blocks of $8 \times 8$ pixels
to exploit the spatial correlations between pixels within the same image. DCT-based implementations (closely related to the Discrete Fourier Transform) are also popular in numerous other coding standards (H.261, JPEG) due to their high decorrelation performance and availability of fast algorithms for real-time implementations.

Motion compensated prediction is a powerful approach used to reduce the temporal redundancies that exist between frames. This prediction technique is also utilized in the proposed MPEG-1 and 2 standards. The concept of motion compensation is based on the estimation of motion between frames. The previous frame, along with the estimated motion between frames is used to predict the current frame. An image is broken up into $16 \times 16$ blocks (i.e. in MPEG-1 and 2) and only one motion vector is estimated for each of these blocks. The resulting motion vectors are usually sent to the receiver along with the prediction error (i.e. the difference between the original image and the motion compensated prediction image). The combination of the two schemes described above (i.e. intraframe DCT coding and temporal motion compensated prediction) are the key elements of the MPEG coding standards and are referred to as hybrid block-based DCT techniques.

Spatio-temporal down-conversion is another compression technique that is commonly used in video image communication scenarios over very low bitrate channels and for real-time transfer of uncompressed video from storage devices such as hard disks and CD-ROM's [2]. Frame rate down-conversion is a process where frames are dropped while spatial down-conversion refers to the subsampling of the 2D lattice in order to reduce the spatial resolution of individual frames. At the receiving end, the missing frames can be approximated and inserted via motion compensated interpolation while the spatial dimensions of individual frames can be recovered by accurate spatial interpolation methods.

Despite the popularity and success of the existing MPEG-1 and 2 coders, these approaches run into a number of serious problems. The conventional block-based DCT schemes fail to adequately model object motion within the scene, and also suffer from visually degrading blocking artifacts. The former case arises when the object motion is non-translatory (i.e. object undergoes rotation, deformation, or zoom) or
when there is multiple motion within an image block. The latter blocking artifacts, refer to the artificial intensity discontinuities at the borders of neighboring blocks within a frame. These are introduced by the independent processing of each block using different quantization strategies. In addition to the problems outlined above, the conventional approaches deal with video exclusively at the frame level, thereby preventing the manipulation of individual objects within the bitstream.

Recently, greater attention has been paid to a newer generation of coding schemes which are object-based [3, 4]. These methods rely on the techniques of image analysis and computer graphics to represent the image signals using their structural features such as contours and regions. Future coding techniques such as these, focus not only on more efficient compression methods but also on providing better ways to represent and exchange visual information. Anticipating this demand, the MPEG group initiated the MPEG-4 standard in 1994 with a mandate to standardize algorithms and tools for coding and flexible representation of audio-visual data for future multimedia applications [5, 6]. The MPEG-4 efforts aim to meet these challenges by addressing four main requirements: (i) compression efficiency, (ii) high interactive functionality, (iii) coding of natural and synthetic data, and (iv) universal accessibility and robustness in error prone environments. A high coding efficiency of audio-visual data continues to be an important component of the MPEG-4 video standard. The standard will provide for an object-layered bit-stream syntax where individual objects within a video sequence are coded into a separate object bit-stream layer. The shape and transparency of each object as well as additional parameters (i.e. scale, location, zoom, etc.) are included in the bit-stream. This is a significant deviation from the block-based approaches taken in MPEG-1 and 2 (MPEG-4 will also support all the functionalities provided by MPEG-1 and 2).

The efficient coded representation just mentioned will also support content-based interactivity that will allow one to use and present multimedia data in a highly flexible manner. This will enable content-based storage and retrieval functionalities (i.e. World Wide Web, multimedia databases, video on demand, etc.) as well as editing and compositing features. MPEG-4 will also assist the efficient coding and repre-
sentation of both, natural video (i.e. pixel-based) and synthetic data. Finally, it is envisioned that the algorithms and tools need to be designed with the awareness that the audio-visual data will be accessed in heterogeneous network environments, possibly under significant error conditions (i.e. mobile channels). The targeted bitrates for MPEG-4 are between 5-64 kb/s for mobile applications and up to 4 Mb/s for TV/film applications [1]. MPEG-4 is expected to be a completed standard by January 1999. Nevertheless, MPEG-7 is already on its way (to be completed by 2001) with the objective of building upon the previous standards and focusing more heavily on the audio-visual content.

1.1 Thesis Objectives and Outline

As mentioned previously, the spatial down-conversion within each frame of a video sequence is a common compression technique used to retain an uncompressed version of the signal at a lower resolution. If this approach is used, then the original, higher resolution image can be approximated by accurate spatial interpolation methods. Image expansion or digital zooming is desirable in many applications, including, the compression scheme just outlined, and image zooming for editing purposes as envisioned in the object-based approaches in MPEG-4 and 7 (i.e. editing of individual objects). It is well known that no new high-frequency information can be generated by linear, shift-invariant interpolation techniques, including ideal band-limited interpolation and its approximations [2]. To this effect, we examine several nonlinear interpolation methods based on both, structural and statistical models for improved definition. The proposed structural models are based on nonlinear filters which have the desirable properties of edge and image detail preservation while the latter statistical approaches include a Gibbs Random Field model which adapts well to both, the textured and smooth areas of the image. These are outlined in the first two chapters of the thesis.

In the two subsequent chapters of the thesis we examine the temporal prediction problem for future object-based video coding environments (as in MPEG-4 and 7), in
light of the problems associated with the block-based methods. We focus our attention on videophone-type sequences which consist of head-and-shoulder type images. Color segmentation schemes are utilized in order to partition an image into a facial area and a set of arbitrarily-shaped regions. These are subsequently represented by triangular mesh-based models for motion compensated prediction purposes. The results are compared to the conventional block-based approaches.

In Chapter 2, the image expansion or superresolution problem is introduced along with the conventional linear interpolation techniques. Nonlinear filters structures are proposed in this chapter for color image interpolation. Digital filters based on the class of order statistic filters are first introduced, followed by a discussion of median-hybrid filters. The latter set of filters are extended to the multichannel case of color images, resulting in the proposed vector FIR median hybrid filters. These nonlinear interpolation filters are applied to different sampling lattices, and the experimental results provide a comparison with the conventional linear schemes.

In Chapter 3, spatial image interpolation is performed using a statistical modeling approach. The Gibbs Random Field model is first introduced using a binary valued energy function. The estimation of the model parameter values are subsequently investigated using a gradient ascent method. An iterative deterministic relaxation algorithm, ICM (Iterative Conditional Mode) is used to obtain the results of the interpolated image. The ICM method is realized using binary logic-type filters for a fast and efficient implementation. This approach is compared to the method of Simulated Annealing, and other linear techniques.

In Chapter 4, we focus our efforts on the automatic location and tracking of the facial region in head-and-shoulders videophone-type sequences using color and shape information. The method we present utilizes the skin-tone distribution of the histograms in the HSV color space to initially extract the skin-like regions. A series of fuzzy membership functions are used to correctly classify and retain the facial area in the case of additional falsely included regions. An aggregation of these features within the framework of a knowledge-based decision system finally provides the mechanism of selecting the facial area from the set of candidate regions. Facial region extraction
is useful for both, video coding purposes and in supporting future content-based functionalities.

In Chapter 5, we present an object-based approach to motion compensated prediction for videophone-type applications. The techniques of image analysis and digital image warping are utilized to predict the current frame from the previous one. The previous frame is first segmented into a facial region and a set of arbitrarily-shaped regions using the color information with suitable constraints. To achieve this, the facial extraction technique from the previous chapter is used along with a general segmentation scheme in the HSV color space. A triangular mesh-based model is subsequently constructed using an appropriate control point selection algorithm. A suitable mesh tracking scheme follows for the assignment of the corresponding motion vectors. An image warping procedure is finally used in order to form the predicted image.

In Chapter 6, the conclusions are drawn along with some insight to future research and development directions.

The following publications have been made as a result of the work in this thesis:

**Book Contributions**


**Journals**


Conferences


Chapter 2

Spatial Image Interpolation Using Nonlinear Filter Structures

2.1 Introduction

The process of downsampling or decimation, is an effective way of reducing the spatial information in a digital image, whether it be for storage or transmission purposes. Conversely, the reverse procedure of this, referred to as interpolation or upsampling, is useful in recovering the original high resolution image from its downsampled version or for simply resizing or zooming a digital image. Decimation and interpolation schemes find their way into many practical applications, such as video compression systems, image/video zooming for editing or display purposes, and standards conversion.

The spatial down-conversion within each frame of a video sequence is a common compression technique used to retain an uncompressed version of the signal at a lower resolution. In this approach, the original, higher resolution image can be approximated by accurate spatial interpolation methods. Video zooming has also been a fundamental component in enabling digital video on the Personal Computer, (PC). PC graphics board manufacturers have incorporated pixel zooming functions into the graphics chips in order to eliminate bandwidth bottlenecks, and permit real-time video playback at increased spatial resolutions [7]. Image zooming is also a significant feature that may be utilized for flexible content-based access and manipulation of in-
dividual objects in MPEG-4 or 7 video sequences. Figure 2.4, (a) illustrates the case where an object (i.e. circular object) is taken from video sequence “A” and is zoomed and incorporated into video sequence “B” by editing and compositing (i.e. assuming that the different video objects are available to the user). In part (b), we depict a compression scheme where an image sequence is transmitted at a lower spatial resolution and then played back at the receiver at an increased resolution using a spatial image interpolation technique. Therefore, the need for robust, high quality interpolation algorithms has become increasingly important. Here, we focus on obtaining a higher resolution image from a single low resolution image which is also known as the intraframe superresolution problem [2]. However, we note that superresolution may also refer to the reconstruction of a high resolution image from a number of lower resolution frames.

Many conventional interpolation techniques have been used to increase the spatial resolution of an image [8], [9], [10]. Some of these include pixel replication, bilinear interpolation, and spline based methods. However, these techniques often perform rather poorly in a subjective sense, as they tend to cause blurring or introduce artifacts in the form of jagged lines or blockiness in the interpolated image. This degradation in image quality is due to the deviations of the above linear filters from the ideal lowpass filter. Other statistical approaches based on Markov Random field models have also been successfully implemented in solving this problem [11], [12]. These techniques require that parameters be estimated for the prescribed underlying image model. However, parameter estimation is a computationally demanding procedure that limits the usefulness of these algorithms for any real-time applications.

The conventional linear schemes described above are well established methods for univariate two-dimensional signals, such as grey level images. An extension of these techniques to multivariate data, such as color images is not straightforward. Processing each color component separately will fail to take into account the correlations between channels (i.e. colors). Moreover, the linear filters suffer from artifacts in the interpolated images. To preserve picture sharpness as well as to provide smoothness, the context of the image, specifically the edge information should be taken into ac-
count in the interpolation process, since the edges convey most of the information for human visual perception. The directional sensitivity or orientational tuning in the human visual system is the dominant perceptual property which enables a human observer to detect faint lines and/or edge patterns in low contrast or noisy images, perceive discontinuous line segments as longer, continuous segments, and many other tasks that are taken for granted [13]. Recent work [14] has indicated that order statistic (OS) based nonlinear filters outperform their linear counterparts in output quality for this univariate case. In this chapter, color image interpolation is performed using a class of nonlinear filters based on vector OS filters. Due to the multichannel nature of color, the multivariate samples are processed as vectors as opposed to component-wise scalars. Two different resampling (i.e. decimation/interpolation) schemes are investigated here, the rectangular, and quincunx lattices. Color images are interpolated from their downsampled images using various Vector FIR-Median hybrid (VFMH) filters, as well as the Vector Median (VMF), and are subsequently compared to the linear component-wise techniques. Experimental results are presented for both real and synthetic images.

2.2 Digital Resampling

Image resampling is the process of transforming a discrete image defined at a certain grid of points to that of a new set of points. This transformation is carried out in the digital domain, although conceptually, it can be thought of as the following two-step process: interpolation of the discrete image to a continuous image and then sampling this interpolated image. This procedure can be used to increase the number of points in an image (termed interpolation or upsampling), or conversely, to reduce the size of an image through a decimation or downsampling process. In the sections below we will consider the alteration of the sampling rate from a digital signal processing point of view for the 1-D case. This can then be extended to the 2-D case for images. A good review on multirate digital signal processing as applied to systems for decimation and interpolation can be found in [15].
2.2.1 Sampling Rate Reduction by an Integer Factor

The sampling rate of a discrete-time signal, \( x[n] \) can be reduced by taking every \( M^{th} \) value of \( x[n] \) thereby yielding a new sequence \( x_d[n] \) (i.e. the downsampled or decimated sequence)

\[
x_d[n] = x[nM]
\]  

(2.1)

The process of downsampling by a factor of \( M \) can be carried out without aliasing if the original sampling rate was at least \( M \) times the Nyquist rate or if the bandwidth of the sequence is first reduced by a factor of \( M \) by discrete-time filtering [16]. The effect of this is more easily seen in the frequency domain.

The discrete-time Fourier transform (DTFT) of the downsampled sequence is related to the DTFT of the original sequence as follows

\[
X_d(e^{j\omega}) = \frac{1}{M} \sum_{i=0}^{M-1} X(e^{j(\omega/M-2\pi i/M)})
\]  

(2.2)

\( X_d(e^{j\omega}) \) can be thought of as being composed of \( M \) copies of the periodic Fourier transform \( X(e^{j\omega}) \), frequency scaled by \( M \) and shifted by integer multiples of \( 2\pi \). Figure 2.5 illustrates the downsampling process [16]. Part a) of the figure shows the Fourier transform of a bandlimited continuous time signal and part b) displays the Fourier transform of the sampled version of \( x_c(t) \) with sampling period \( T \). In part c) there is a normalization of the frequency axis (i.e. going from the impulse train \( x_c(nT) \) to the sequence \( x[n] \) introduces a time normalization) while part d) shows the DTFT of the downsampled sequence when \( M = 2 \). Finally, part e) shows the downsampled sequence plotted as a function of the continuous-time frequency variable \( \Omega \) (i.e. once again a relabeling of the frequency axis). Thus, we see in this example that the original sampling rate is exactly twice the minimum rate to avoid aliasing (i.e. \( 2\pi/T = 4\Omega_N \)). In this case, when the original sampled sequence is downsampled by a factor of \( M = 2 \) no aliasing results. However, if this factor is greater than 2, then aliasing will result. In general, to avoid aliasing in downsampling by a factor of \( M \) requires that \( \omega_N M < \pi \). If this condition does not hold, aliasing occurs unless
$x[n]$ is filtered by an ideal lowpass filter with cutoff frequency $\pi/M$. Prefiltering, however, often removes some of the higher frequencies of the original signal. A general system for downsampling is shown below.

![Figure 2.1: General system for sampling rate reduction by M.](image)

### 2.2.2 Increasing the Sampling Rate by an Integer Factor

More frequently, it is desired to increase the sampling rate of a signal. Upsampling or interpolation is demonstrated in Figure 2.2. The system on the left part of the schematic is called a sampling rate expander where

$$x_e[n] = \begin{cases} x[n/L], & n = 0, \pm L, \pm 2L, \ldots, \\ 0, & \text{otherwise} \end{cases} \quad (2.3)$$

or

$$x_e[n] = \sum_{k=-\infty}^{\infty} x[k]\delta[n - kL] \quad (2.4)$$

![Figure 2.2: General system for sampling rate increase by L.](image)
The output of the expander is then fed to a lowpass discrete-time filter with cutoff frequency $\pi/L$ and gain $L$. Described in a simple manner, interpolation can be performed by inserting $L - 1$ zeros between samples of the original sequence $x[n]$ and then lowpass filtering with a cutoff frequency of $\pi/L$ and gain $L$. Once again, let us have a look at the effect of this process in the frequency domain. Taking the Fourier transform of equation (2.4) it can easily be shown that

$$X_e(e^{j\omega}) = X(e^{j\omega L})$$

(2.5)

that is, the Fourier transform of the output of the expander is a frequency-scaled version of the Fourier transform of the original sequence (i.e. $\omega$ is replaced by $\omega L$). Figure 2.6 illustrates the complete interpolation process [16]. Figure 2.6 shows the Fourier transform of a bandlimited continuous-time signal while part b) depicts the discrete-time Fourier transform of the sequence $x[n] = x_c(nT)$ where $\pi/T = \Omega_N$. In part c) the output of the expander $X_e(e^{j\omega})$ is illustrated with $L = 2$ (i.e. according to equation (2.5)). Finally, part e) displays the desired interpolated signal $x_i[n]$. This is obtained by filtering the expander output with the ideal lowpass filter $H_i(e^{j\omega})$ indicated in part d).

From the discussion above, it is evident that a certain amount of distortion in the interpolated signal is inevitable, since in practice ideal lowpass filters cannot be implemented exactly. The interpolated signal described above is usually obtained by convolving the impulse response of the lowpass filter (which will be referred to as the interpolating function or kernel) with the expander output. Conventional interpolation schemes typically employ finite-impulse response (FIR) filters to carry out this task.

The interpolation process for the 1-D case above can be extended and applied to 2-D signals such as images. We must note however that practical applications involving display devices such as monitors, form digital images on rectangular sampling grids. For this reason, we will restrict ourselves to rectangular sampling geometries and any sampling rate alteration of an image will take place in the horizontal and/or
vertical directions only. The steps which were carried out above to upsample in 1-D are followed below in order to demonstrate the process for 2-D signals. An image is upsampled by a factor of $L_1$, and $L_2$, in the horizontal and vertical directions, respectively, by first inserting $L_1 - 1$ zeros between columns and $L_2 - 1$ zeros between rows. This is followed by lowpass filtering with the ideal 2-D filter $H_i(e^{j\omega_1}, e^{j\omega_2})$ as done previously. If we apply the general system of Figure 2.2 to the 2-D case then the expander output of equation (2.4) becomes

$$x_e[n_1, n_2] = \sum_{k_1=-\infty}^{\infty} \sum_{k_2=-\infty}^{\infty} x[k_1, k_2] \delta[n_1 - k_1 L_1] \delta[n_2 - k_2 L_2] \tag{2.6}$$

The resulting Fourier transform of this sequence is once again a frequency-scaled version of the original sequence in the two orthogonal directions $\omega_1$, and $\omega_2$ (i.e. the horizontal and vertical directions)

$$X_e(e^{j\omega_1}, e^{j\omega_2}) = X(e^{j\omega_1 L_1}, e^{j\omega_2 L_2}) \tag{2.7}$$

Therefore, the illustrations of Figure 2.6 can be viewed as slices or cross-sections of the corresponding 2-D Fourier transforms in the horizontal or vertical directions. Distortion in the upsampled image does not occur if the appropriate frequency-scaled images are removed in all of the horizontal and vertical slices of the 2-D plane.

When dealing with 2-D signals such as images, significant computational savings can be realized if the 2-D interpolation functions are chosen to be separable extensions of the 1-D functions, that is $h[n_1, n_2] = h_1[n_1] h_2[n_2]$. In this case, the 2-D convolution for image resizing can be done as a 1-D convolution along each column of the image to form an intermediate image followed by a 1-D convolution along each row of the intermediate image to form the final image [17]. For an $M \times M$ FIR filter, a separable convolution results in $2M$ multiplications per pixel as opposed to $M^2$ for the non-separable case.

As a final remark, we may also note that the sampling rate of a sequence can also be changed by a non-integer factor. This operation can be performed by combining the
concepts of decimation and interpolation. By choosing $L$ and $M$ we can approach arbitrarily close to any desired ratio of sampling periods. Theoretically, this approach may be taken for any ratio, however, ones that require large values of $L$ and $M$ may not be practically feasible.

2.3 Conventional Interpolation Methods

In the previous section it was shown that an image could be resized to one of a larger size (by a factor of $L$) by inserting $L - 1$ zeros between sample points and subsequently lowpass filtering. Implementing the lowpass filter in the frequency domain demands that the entire image is processed at once. This requires that the image is read into memory, followed by a Fast Fourier Transform (FFT) computation, then an ideal filtering operation, and finally carrying out an Inverse Fast Fourier Transform (IFFT). This is an expensive and computationally prohibitive approach for real-time video scaling. For example, one of the advanced families of dedicated digital signal processing (DSP) chips from Texas Instruments is capable of performing a 256 point FFT in 13.3 $\mu$s [18]. If we were to scale a video sequence to full screen at a 1024 x 1024 resolution, then the FFT and IFFT calculations alone would require approximately 110 ms. However, full motion video at 30 frames/s requires at most a delay of approximately 33 ms between frames. Thus, we run into cost and performance issues in frequency domain processing.

In the spatial domain, the image can be processed sequentially in a raster scan fashion allowing for a practical implementation. However, the ideal lowpass filter in the spatial domain corresponds to an infinite extent sinc function. Since this is not realizable, many interpolating functions with finite impulse response (FIR) filters of short duration have been employed to best approximate the ideal lowpass filter.

The easiest interpolation algorithm to implement from a computational standpoint is the nearest neighbor (or zero order hold) algorithm where each pixel is given the value of the sample which is closest to it. In this technique, the sampled image is convolved with a rectangle function which is equivalent to multiplying the image in
the frequency domain by a sinc function. However, the sinc function is quite a poor lowpass filter due to its large sidelobes. Thus, if an image is resized to a larger size, the nearest neighbor algorithm will fail to perform well since the pixel values are merely replicated and it will give the resampled image a blocky appearance.

A slightly more complicated interpolation technique is the bilinear algorithm. With linear interpolation, each new sample is determined by assuming that it falls on a straight line connecting the two surrounding samples. In the 2-D case, the sample is evaluated by using a linear combination of the four closest pixels. This can also be expressed as a convolution of the sampled signal with a triangle function. This corresponds to a slightly better approximation to the ideal lowpass filter than the previous algorithm, however, it tends to attenuate frequencies near the cutoff frequency (i.e. aliasing some of the frequencies above the Nyquist rate into those below it). As a result, the interpolated image tends to become smoothed, especially near the edges.

More complex interpolating functions such as cubic B-splines have also been investigated. Hou and Andrews examine the use of cubic B-splines quite thoroughly in [9]. The nearest neighbor algorithm interpolates on the basis of a single point for the 1-D case; the linear case on the basis of two points while the cubic B-spline utilizes four points. Once again, the complexity of the algorithm increases but it also improves as a lowpass filter. The cubic B-spline is a piecewise continuous function that is positive symmetric and can basically be formed by four convolutions of the simple rectangular function. The frequency domain spectrum of this function indicates that it has good stopband behavior but tends to smooth out the higher frequencies just below the cut-off.

Another interpolation scheme that uses four data samples to compute the interpolated values is the cubic convolution kernel [10]. This is again a piecewise continuous third-order function which belongs to the general class of cubic spline functions. The function is symmetric but unlike the B-spline it dips below zero between the outer sample points. The frequency domain response indicates a better high-frequency performance than the cubic B-spline due to a better amplification of the frequencies just
below the cut-off frequency. Figure 2.7 displays the normalized frequency response (i.e. magnitude of the Fourier Transform) of the conventional interpolating functions that are commonly utilized. The dashed box gives an idea of how each function compares to the ideal lowpass filter. The cubic convolution kernel (i.e. cubic spline) appears to give the best results of the four methods by having the best passband and stopband performance at, however, the expense of a higher computational complexity. Once again it is worth mentioning that most algorithms are implemented in a separable fashion as we have illustrated here so that the number of arithmetic operations are reduced. In this case, the convolution is performed as a series of 1-D convolutions along each row and each column of the image as was stated in the previous section. Higher order interpolating functions can also be to used to better approximate the ideal lowpass filter at the expense of an increase in the computational complexity. However, a larger size interpolating kernel also tends to degrade the picture quality near the borders of the image.

An important limitation of most available acquisition devices today (i.e. still frame or video cameras) is that the digital images are recorded at a lower resolution than desired. This is related to the physical limitations of the image sensors such as finite cell area and finite aperture time. Most images contain sharp edges that are not strictly bandlimited. As a result, digital images usually suffer from loss of high frequency detail due to low resolution sensor point spread functions, aliasing due to undersampling, and possible optical blurring from motion or out-of-focus. It is well known that no new high frequency information can be generated by linear, shift-invariant interpolation techniques, including ideal band-limited interpolation (i.e. ideal lowpass filter) and its approximations [2]. Thus, we explore the problem of spatial interpolation based on the class of nonlinear order statistic filters.

2.4 Order Statistic Filters

Digital filters based on order statistics have found extensive applications in many areas of image processing, such as image analysis, restoration, enhancement, and
noise removal [19]. The theoretical basis for this class of filters lies in the theory of robust statistics [20], [21]. Robust estimation aims to find the parameters that best fit the bulk of the data and to identify and reject the outlying observations. This is particularly important in image filtering, so that the edge information and image details are preserved while the noise is suppressed. This OS class includes many nonlinear filters, such as the median filter [22], [23], the α-trimmed mean [24], and the median hybrid filter [25], to name a few. The appropriate filter is chosen according to the specific application and the characteristics of the problem at hand.

OS filters are based on the ordering of scalar data values for the univariate case. A generalization of the filtering process consists of the following three steps: i) windowing, ii) algebraic ordering, iii) and linear weighting. In an image \( f \) (consisting of univariate values), a section of the image is first selected by a window \( W \), yielding \( W = \{f_1, f_2, \ldots, f_n\} \). The next step involves the algebraic ordering of these values, which results in the following \( W = \{f_{(1)}, f_{(2)}, \ldots, f_{(n)}\} \), where the \( f_{(i)} \), \( i = 1, 2, \ldots, n \) are the ordered samples. Finally, these ordered values are weighted appropriately and summed. This process can be described mathematically by the expression \( g_i = \sum_{j=1}^{n} a_j f_{(j)} \) where \( f_{(j)} \) are the ordered samples, \( a_j \) are the filter coefficients, and \( g_i \) is the filter output. Different filters can be realized by properly selecting the filter coefficients. The following choices of coefficients result in the median, and the α-trimmed mean filters, respectively

\[
a_j = \begin{cases} 
1 & j = \nu + 1 \\
0 & j \neq \nu + 1
\end{cases}
\]

\[
a_j = \begin{cases} 
\frac{1}{n(1-2\alpha)} & j = \alpha n + 1, \ldots, n - \alpha n \\
0 & \text{otherwise}
\end{cases}
\]

where \( n = 2\nu + 1 \). A large number of filters can be realized with this structure. The performance of each filter in this OS class depends on several factors some of which include, image preservation characteristics, edge response, noise removal, and computational complexity. As mentioned earlier, the proper selection of a suitable
filter depends on the specific image processing application.

2.4.1 Median Hybrid Filters

The median is the best known filter from the family of order statistics. Its widespread use is due to its simplicity, computational speed, edge preservation, and impulse removal properties. The median filter has been studied quite thoroughly and it exhibits very well known and desirable statistical [22] and deterministic (i.e. root signals which are invariant under median filtering) properties [23]. However, it has the undesirable effect of creating linear streaks or blotches in the filtered output image. Fine details, sharp corners, and narrow lines are also destroyed, because the ordering process in median filtering destroys any structural and spatial neighborhood information. This is extremely undesirable in image interpolation, since fine details, and image edges contain high frequency content that carry very important information for human visual perception. The median hybrid filter (FMH) has been shown to improve upon the image detail preservation properties of the median filter. In addition to this, the FMH filter performs well in attenuating noise, and preserving step edges.

Median hybrid filters are a combination of linear filters and median filters. Linear subfilters are used to take into account the spatial structure of the input image. The median hybrid filter is defined as follows

\[ g_i = \text{Median} \left( \Phi_1(f_i), \ldots, \Phi_n(f_i) \right) \]  

(2.8)

where \( \Phi_j(f_i), j = 1, \ldots n \) are linear FIR or IIR filters. The flexibility in this structure allows one to design the linear filters in such a way that fine horizontal, vertical, and/or diagonal lines can be preserved in an image. By using a small number of FIR subfilters of the averaging type, the number of sorting and multiply operations can also be greatly reduced [25]. This offers an improvement in speed over the median filter, while at the same time preserving the fine image details. Nieminen et al. report that the computation time of a multilevel detail preserving FMH filter is over two times less than that of a median filter, and over seven times less than a K-nearest
neighbor averaging filter in a $5 \times 5$ window [26].

An important property of FMH filters is the existence of root signals, where input sequences are invariant to repeated filtering operations. Roots signals can give an indication of the filter’s ability to preserve fine lines and details when applied to an input image. In [26], a test image of a thin circular ring was generated and used to analyze the effects of filtering with various FMH filters, the median, and the K averaging filter. When a very thin ring was used (i.e. width of 2 pixels), only an FMH filter with subfilters in all eight orientations did not change the input image upon filtering. All other filters altered the ring image. The median filter was not able to preserve the original image due to the small width of the ring while the averaging filter distorted the signal levels of the ring. Thus, this preservation property of the FMH filter is highly desirable.

The statistical properties of FMH filters have been analyzed for many important classes of input signals and noise distributions. A statistical analysis by computer simulation indicates that the FIR-Median hybrid filters are able to preserve edges in noisy images better than the median and the K-nearest neighbor averaging filter [26]. This edge preservation property is also important in retaining sharp, filtered output images.

In image interpolation, four factors are of particular importance: i) edge preservation, ii) fine detail preservation, iii) unbiasedness (i.e. directional or illumination bias), iv) and computational complexity. FIR-Median filters perform strongly in all four of the above mentioned areas. In [27] a comprehensive list of various nonlinear filters and their performance for various figures of merit are conveniently summarized. The tabulated results also indicate that the FMH filters are strong performers in all four areas indicated above, which is critical to the interpolation process.

2.5 Multichannel Data

Multichannel or multispectral signal processing has been a rapidly growing area of interest recently, primarily due to the numerous applications. The advances in high
resolution, true-color graphics cards along with high scanning monitors, and state of the art active matrix TFT screens have all demanded an increased attention in color image processing techniques. However, the multichannel nature of color adds an increased complexity when processing color images. Data storage is multiplied by a factor of three and computational complexity also increases over the monochrome case. This is due to the fact that a color image is a three channel, two dimensional signal (i.e. each pixel is a 3D vector composed of red, green, and blue additive components). In addition to this, a strong correlation exists between the different channels which suggests that a vector approach be taken in order to utilize the interchannel correlations [28]. Vector processing has been successfully applied recently, in many areas of image processing such as filtering [29], [30], enhancement [31], edge detection [32], [33], and restoration [34], [35]. Transformation techniques such as the Karhunen-Loeve transform have also been used to decorrelate the three channels so that monochrome methods could be applied to each of the decoupled channels [36]. This is also a valid approach, however, multichannel signal processing techniques appear to be a more natural approach to the problem [35]. Thus, univariate order statistic filters are extended to the case of multichannel data for color image interpolation.

2.5.1 Ordering of Multivariate Data

In the univariate case, OS filters are based on the ordering of scalar data values and an extension to multivariate data is not straightforward. As mentioned earlier, color images comprise an important class of multivariate signals and therefore an appropriate framework must be chosen for processing them (i.e. in order to take advantage of the interchannel correlations of the RGB color planes). In the case of color, each data sample is a vector value with red, green, and blue components. Let the set of \( n \) vectors be within a window \( W \), where \( W = \{ f_1, f_2, \ldots, f_n \} \) and each vector, \( f_i = [R_i, G_i, B_i] \) be a point in the RGB color space. The ordering of this multivariate data can be performed according to the following sub-ordering principles: Marginal ordering, reduced ordering, partial ordering, and conditional ordering [37]. In reduced
ordering (R-ordering), the vectors, \( f_i \) are ranked according to some distance criteria. This reduces multivariate ranking to a simple scalar ranking operation on a set of distance values and retains the color component correlations. For this reason, we restrict our attention to this ordering scheme. In R-ordering, the distance values are computed for each vector in the window as follows

\[
d_i = \sum_{j=1}^{n} \| f_i - f_j \|, \quad i = 1, 2, \ldots, n
\]  

(2.9)

using an appropriate vector norm \cite{38}. Here we use the Euclidean distance due to its effectiveness and simplicity. Using this expression, one can compute an associated set of distances \( \{d_1, d_2, \ldots, d_n\} \) for the set \( \{f_1, f_2, \ldots, f_n\} \). The scalar ranking of these distances yields the ranked vector set \( \{f_{(1)}, f_{(2)}, \ldots, f_{(n)}\} \), where the vector \( f_{(i)} \) is the \( i^{th} \) order statistic. This ranking process is the basis of the nonlinear Vector Median type filters described below.

### 2.6 Vector Median Type Filters

A natural extension of the median filter to the multichannel case is the vector median (VMF). The VMF filter is known to exhibit properties similar to those of the scalar median, that is, the preservation of edges, the existence of root signals, and the suppression of impulsive noise \cite{30}. Using the notation from the previous section, the vector median can be defined as follows

\[
f_{VM} = f_{(1)} = VM \{f_1, f_2, \ldots, f_n\}
\]  

(2.10)

where VM is the vector median operation. The vector \( f_{VM} \) is the one whose distance to all other vectors is a minimum and is also contained in the set of vectors within the sliding window \( W \)

\[
\sum_{j=1}^{n} \| f_{VM} - f_j \|_2 \leq \sum_{j=1}^{n} \| f_i - f_j \|_2 \quad \forall \ i \in \{1, \ldots, n\}
\]  

(2.11)
In a similar way, the FMH filters described earlier can be extended to the multichannel case using Vector FIR-Median hybrid (VFHM) filters. These are defined in an analogous manner

\[ f_{VFHM} = VM \{ \Phi_1, \Phi_2, \ldots, \Phi_n \} \]

where \( \Phi_i \) are multichannel, linear FIR filters. The VFHM filters are shown to have good edge preservation properties and good noise attenuation [39]. Once again, the flexibility of the linear FIR substructures allows one to reduce the computation time (by choosing a small number of linear subfilters), and preserve fine details of the image by selecting the \( \Phi_i \) appropriately. Furthermore, the VFHM filters preserve the edge shape and location for each signal component simultaneously by operating vectorially and utilizing the interchannel correlations. Component-wise filtering on the other hand, may cause edge shifts (i.e. known as edge jitter) in the different channels which may result in new, unwanted colors in transition areas where the original color is rapidly changing. In [39], the VFHM filter is shown to outperform the component-wise FMH filter near a simulated step edge. Clearly, the VFHM filters described above, possess many desirable properties that make them suitable for image interpolation.

### 2.6.1 Rectangular and Quincunx Resampled Lattices

The interpolation of a one-dimensional signal by a factor of \( L \) requires that for each known sample value, \( L - 1 \) new samples be determined and inserted in the sequence. The placement of these newly determined values is straightforward in the 1-D case, however, in the two-dimensional case of images, the spatial arrangement of these values can be quite arbitrary. This arrangement of pixel values forms a constellation or lattice of points. Two practical cases are examined here, that of rectangular and quincunx lattices. The original image is decimated (without prefiltering), and then interpolated back to its original size so that quantitative as well as qualitative comparisons can be made. This procedure is outlined below.

In two-dimensional decimation of images, a subset of the original 2-D sequence is
retained and a sub-lattice of sampled points is formed. This lattice can be determined from the expression \( g[n] = f[Mn] \) for an \( M \) -fold decimation process, where \( M \) is a \( 2 \times 2 \) nonsingular matrix of integers, \( n = (n_1, n_2) \) (\( n_1 \) and \( n_2 \) are the horizontal and vertical spatial indices, respectively, in the 2-D integer plane), \( f[\cdot] \) is the original image, and \( g[\cdot] \) is the output or subsampled set of points [40]. A number of different lattices can be generated by properly choosing the matrix \( M \). The decimation ratio, or equivalently the compression ratio is found by taking the absolute value of the determinant of the matrix \( M \) (i.e. \( |\det M| \)). The rectangular and quincunx subsampled lattices are shown in Figure 2.3 below. If the \( 5 \times 5 \) sample of points in the figure represent a particular section of an image, then only the points \( X \) are retained from the decimation process in each of the schemes mentioned above. Subsequent interpolation of the image from its decimated version requires that the missing pixels \( Y \), and \( Z \) be determined, so that the regenerated image is as close as possible to the original. In the case of linear interpolation, the missing pixels are replaced with zeros and then this zero-interlaced image is subsequently lowpass filtered.

\[
\begin{array}{cccccc}
X & Y & X & Y & X \\
Y & Z & Y & Z & Y \\
X & Y & X & Y & X \\
Y & Z & Y & Z & Y \\
X & Y & X & Y & X \\
\end{array}
\]

(a)

\[
\begin{array}{cccccc}
X & Y & X & Y & X \\
Y & X & Y & X & Y \\
X & Y & X & Y & X \\
Y & X & Y & X & Y \\
X & Y & X & Y & X \\
\end{array}
\]

(b)

Figure 2.3: (a) Rectangular decimation (b) Quincunx decimation.
2.7 Image Interpolation Using Vector-FIR Median Hybrid Filters

Several VFMH filters were used for each of the decimation schemes described previously. In rectangular decimation, the pixels $Y$, and $Z$ must be interpolated using the developed nonlinear filters. A subsection of the rectangular lattice of Figure 2.3 (a) is shown here to illustrate the six different VFMH filtering schemes that were implemented.

\[
\begin{array}{ccc}
X_1 & Y_1 & X_2 \\
Z_2 & Y_2 & Z_1 & Y_3 \\
X_3 & Y_4 & X_4 \\
\end{array}
\]

1. $\text{VFMH}_{R1}$:

$Z$ pixels $\Rightarrow$

$Z_1 = \text{VM} \{X_1,X_2,X_3,X_4,X_{FIR1}\}$

$Y$ pixels $\Rightarrow$

$Y_1 = (X_1+X_2)/2$, $Y_2 = (X_1+X_3)/2$

$Y_3 = (X_2+X_4)/2$, $Y_4 = (X_3+X_4)/2$

2. $\text{VFMH}_{R2}$:

$Z$ pixels $\Rightarrow$

$Z_1 = \text{VM} \{X_1,X_2,X_3,X_4,X_{FIR1}\}$

$Y$ pixels $\Rightarrow$

$Y_2 = \text{VM} \{X_1,Z_1,Z_2,X_3,X_{FIR2}\}$
3. \( \text{VFMH}_R^3 \): 

\[
\begin{align*}
Z \text{ pixels } & \Rightarrow \\
Z_1 &= \text{VM} \{X_1, X_2, X_3, X_4, X_{FIR1}\}
\end{align*}
\]

\[
\begin{align*}
Y \text{ pixels } & \Rightarrow \\
Y_2 &= \text{VM} \{X_1, Z_1, Z_2, X_3, X_H, X_V, X_{FIR2}\}
\end{align*}
\]

4. \( \text{VMF}_R \):

\[
\begin{align*}
Z \text{ pixels } & \Rightarrow \\
Z_1 &= \text{VM} \{X_1, X_2, X_3, X_4\}
\end{align*}
\]

\[
\begin{align*}
Y \text{ pixels } & \Rightarrow \\
Y_2 &= \text{VM} \{X_1, Z_1, Z_2, X_3\}
\end{align*}
\]

An additional three filters, \( \text{VFMH}_R^4\), \( \text{VFMH}_R^5\), and \( \text{VFMH}_R^6\), can also be realized by replacing the expression for \( Z_1 \), in \( \text{VFMH}_R^1\), \( \text{VFMH}_R^2\), and \( \text{VFMH}_R^3\) above, by \( Z_1 = \text{VM} \{X_1, X_2, X_3, X_4, X_{D1}, X_{D2}, X_{FIR1}\} \), respectively. In the expressions above, \( X_{D1} = (X_1 + X_4)/2 \), \( X_{D2} = (X_2 + X_3)/2 \), \( X_H = (Z_1 + Z_2)/2 \), \( X_V = (X_1 + X_3)/2 \) and \( X_{FIR1}, X_{FIR2} \), are 12 point, lowpass, FIR filters (4 central multipliers of amplitude 1/2, and 8 outer multipliers of amplitude -1/8). One may also note that the filter structures of methods 2, 3, 5, and 6 are recursive. The Vector Median filter (\( \text{VMF}_R \)) was also applied for comparison purposes. However, median filtering as was mentioned earlier, tends to destroy fine image details [19], and thus, the median hybrid filters should be preferred. All of the Y, and Z pixels in Figure 2.3 (a) can be computed by implementing the non-recursive, and recursive filter schemes in one and two passes over the image, respectively. Similar structures have been applied successfully using the scalar median for the case of univariate data [14].
In the quincunx subsampled lattice of Figure 2.3 (b), five nonlinear interpolators were examined. The filter definitions below refer to the following $3 \times 3$ subsection of this lattice.

\[
\begin{array}{ccc}
Y & X_1 & Y \\
X_2 & Y_1 & X_3 \\
Y & X_4 & Y
\end{array}
\]

1. \(VFMH_{Q1}\):

\[Y \text{ pixels} \Rightarrow Y_1 = VM \{X_1, X_2, X_3, X_4, X_{AVE}\}\]

2. \(VFMH_{Q2}\):

\[Y \text{ pixels} \Rightarrow Y_1 = VM \{X_1, X_2, X_3, X_4, X_{FIR}\}\]

3. \(VFMH_{Q3}\):

\[Y \text{ pixels} \Rightarrow Y_1 = VM \{X_1, X_2, X_3, X_4, X_H, X_V, X_{FIR}\}\]

4. \(VFMH_{Q4}\):

\[Y \text{ pixels} \Rightarrow Y_1 = VM \{X_1, X_2, X_3, X_4, X_{HFIR}, X_{VFIR}, X_{FIR}\}\]
5. VMF_Q:

\[ Y \text{ pixels} \Rightarrow \]
\[ Y_1 = \text{VM} \{X_1, X_2, X_3, X_4\} \]

where \( X_{AVE} = (X_1 + X_2 + X_3 + X_4)/4 \), \( X_H = (X_2 + X_3)/2 \), \( X_V = (X_1 + X_4)/2 \) and \( X_{H FIR}, X_{V FIR} \), are 4 point horizontal and vertical FIR filters (2 central multipliers of amplitude 1 and 2 outer multipliers of amplitude -1/2), respectively, and \( X_{FIR} \), is a 12 point FIR filter (same multipliers as previous 12 point filters). All of the Y pixels can be computed in a similar fashion to \( Y_1 \). The various FIR subfilters illustrated in both schemes above are direction sensitive (i.e. the masks of the filter are in various orientations - horizontal, vertical, diagonal, and/or lowpass) and have been chosen in order to preserve the edges, lines, and fine details of the image.

2.8 Experimental Results

The nonlinear interpolating filters outlined in the previous section were applied to various color images, and the visual results of two are shown below, namely, a 512 x 480 color image of Lena, and a 512 x 480 synthetic image composed of several colors in different orientations (vertical, horizontal, and diagonal). Three linear schemes were also used for comparison purposes. In the rectangular lattice, a resampling (downsampling/upsampling) ratio of 16 was used (i.e. 4 in each direction) while in quincunx, a ratio of 2 was employed. The mean square error (MSE) and the mean absolute error (MAE) criteria were used to compare the results quantitatively and are defined as follows

\[
\text{MSE} = \frac{1}{MN} \sum_{i=0}^{M} \sum_{j=0}^{N} \left\| f_{ij} - \hat{f}_{ij} \right\|_2^2 
\]

(2.13)

\[
\text{MAE} = \frac{1}{MN} \sum_{i=0}^{M} \sum_{j=0}^{N} \left\| f_{ij} - \hat{f}_{ij} \right\|_1
\]

(2.14)
where $M, N$ are the image dimensions, $f_{ij}$ is the vector value of the pixel $(i, j)$ of the original image, $\hat{f}_{ij}$ is the vector value of the pixel $(i, j)$ of the interpolated image, and $\| \|_2, \| \|_1$, are the $L_2$, and $L_1$ vector norms, respectively.

A summary of the results for the two images mentioned above are tabulated in Tables 2.1 and 2.2. The results indicate that most of the nonlinear interpolation methods outperform their linear counterparts. The results are fairly consistent in each case, that is, overall the $VFMHR_6$, and $VFMH_{Q3}$, methods appear to be the best in the two resampling schemes. The bilinear method seems to perform better in a quantitative sense than the other linear schemes. One must note, however, that conventional quantitative measures such as the MSE and MAE are often poor indicators of the perceptual results. In the end, the perceptual results are the ultimate test. Nevertheless, we include the mathematical measures as standard practice. The nonlinear interpolation techniques also perform much better than the conventional linear ones from a perceptual point of view as shown in Figures 2.8 and 2.9. The interpolated images derived from the linear schemes have blocky, jagged lines, and are blurred, while the nonlinear methods appear to somewhat suppress these artifacts, preserve the edges, and retain the image details better.

Figures 2.8 (a) and (b) illustrate the original Lena image (512 × 480 pixels) and its downsampling version (i.e. by 4 in each direction) without prefiltering, respectively. In part (c), the pixel replication method demonstrates the blocky, pixelated output. The bilinear method in part (d) improves upon the pixel replication technique, however, it results in aliasing (i.e. jagged lines) and blurring of the output image. The final linear scheme is shown in part (e), which is the cubic spline. This has reduced aliasing artifacts, but suffers from severe blurring. The results obtained from the linear schemes are consistent with the filters outlined in Figure 2.7. In part (f), the output from the FIR median hybrid ($FMHR_6$) filter (i.e. using the filter structure of $VFMHR_6$ separately in each color plane) is shown. This scheme does not perform as well as its vector counterpart in part (j). It has greater aliasing effects due to possible edge shifts by neglecting the interchannel correlations. The vector median filter (VMF) in part (g) also produces unsatisfactory results as it loses image detail as expected. The
VFMHR_{R2} filter in part (h) is the worst performer of the six proposed nonlinear filters while parts (i) and (j) demonstrate the two best performing interpolators, VFMHR_{R4} and VFMHR_{R6}, respectively. Clearly, VFMHR_{R6} produces the best perceptual results from all of the schemes.

Figure 2.9 displays the results of the various techniques applied to the synthetic image. The original and downsampled image are shown in parts (a), and (b), respectively. The blockiness of the pixel replication method, and the aliasing of the bilinear technique in parts (c), and (d), respectively, are even more apparent in this case. The bluriness is evident in the cubic spline of part (e), while the edge shifts in the FMHR_{R6} scheme of (f) are noticeable. Once again, the VFMHR_{R6} in part (j) outperforms all of the schemes from a perceptual point of view. The VFMHR_{R6} filter performs well near step edges, and preserves the interchannel color correlations. The strong performance of the VFMHR_{R6} filter can be attributed to: i) its FIR subfilters which consist of low-pass filters, as well as masks in the diagonal, horizontal, and vertical orientations, ii) and its processing of the multichannel data in a vectorial fashion which preserves color correlations near signal transitions. One may also note a performance improvement in going from method 1 to 4, 2 to 5, and 3 to 6. Once again, this is due to the addition of the diagonally oriented FIR subfilters in the determination of the pixels, Z_i.

Consistent results were obtained for both of the images in the figures (and resampling schemes) as well as other tested images. A summary of the quantitative results in Table 2.3 also confirms the effectiveness of the nonlinear schemes with other sample images. The computational complexity of the VFMHR_{R6} filter is the highest of the nonlinear methods, however, it is only marginally more complex than the cubic spline method. This increased complexity comes at the expense of having a larger number of subfilters in the window of the vector median operation. Nevertheless, these nonlinear interpolators are quite feasible for implementation. These structures can be designed in hardware as one functional unit of the graphics chip or as a separate ASIC that can provide hardware assisted interpolation on a board level solution (i.e. capture and/or video display device, or HDTV applications).
2.9 Conclusions

In this chapter, spatial interpolation of color images was investigated in order to obtain higher resolution images from acquired lower ones. Nonlinear filtering techniques based on vector order statistics were examined for this purpose. Vector FIR-Median hybrid filters were selected from this OS class of filters due to their robustness, preservation of edge information and image details, and their ability to exploit the existing correlation between the RGB color planes. Several VFMH filters were implemented and compared to the conventional linear techniques using both real and synthetic images. The interpolated images were determined from their downsampled versions for two different decimation schemes, that of rectangular and quincunx decimation. Experimental results indicated that the VFMH filters performed better, both, quantitatively, and aesthetically than the linear methods, and when each channel was processed independently. The former filtering techniques reduce the effects of aliasing (blockiness, jagged lines, and blurring), preserve the step edges and image details better than their counterparts, and retain the correlations between colors by operating vectorially. In addition to this, the computational complexity of these nonlinear filters is on the same order, or slightly higher than their linear counterparts which makes them attractive for implementation in hardware.
Table 2.1: A quantitative comparison of the different interpolation methods using rectangular decimation for Lena and the synthetic image

<table>
<thead>
<tr>
<th>Method</th>
<th>“Lena” MSE</th>
<th>“Lena” MAE</th>
<th>Synthetic Image MSE</th>
<th>Synthetic Image MAE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pixel Replication</td>
<td>1098.2</td>
<td>30.44</td>
<td>1377.3</td>
<td>9.94</td>
</tr>
<tr>
<td>Bilinear</td>
<td>363.5</td>
<td>19.16</td>
<td>743.8</td>
<td>10.25</td>
</tr>
<tr>
<td>Cubic spline</td>
<td>423.6</td>
<td>21.72</td>
<td>833.1</td>
<td>10.25</td>
</tr>
<tr>
<td>VFMHR1</td>
<td>361.0</td>
<td>18.94</td>
<td>744.1</td>
<td>9.21</td>
</tr>
<tr>
<td>VFMHR2</td>
<td>410.9</td>
<td>20.32</td>
<td>772.6</td>
<td>8.70</td>
</tr>
<tr>
<td>VFMHR3</td>
<td>364.2</td>
<td>19.15</td>
<td>761.5</td>
<td>8.84</td>
</tr>
<tr>
<td>VFMHR4</td>
<td>351.4</td>
<td>18.71</td>
<td>726.1</td>
<td>9.59</td>
</tr>
<tr>
<td>VFMHR5</td>
<td>359.5</td>
<td>18.9</td>
<td>727.5</td>
<td>9.58</td>
</tr>
<tr>
<td>VFMHR6</td>
<td>347.1</td>
<td>18.57</td>
<td>727.5</td>
<td>9.58</td>
</tr>
<tr>
<td>FMHR6</td>
<td>355.4</td>
<td>18.75</td>
<td>743.6</td>
<td>9.81</td>
</tr>
<tr>
<td>VMFR</td>
<td>420.4</td>
<td>20.61</td>
<td>769.6</td>
<td>10.51</td>
</tr>
</tbody>
</table>

Table 2.2: A quantitative comparison of the different interpolation methods using quincunx decimation for Lena and the synthetic image

<table>
<thead>
<tr>
<th>Method</th>
<th>“Lena” MSE</th>
<th>“Lena” MAE</th>
<th>Synthetic Image MSE</th>
<th>Synthetic Image MAE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pixel Replication</td>
<td>172.71</td>
<td>9.70</td>
<td>142.21</td>
<td>1.01</td>
</tr>
<tr>
<td>Bilinear</td>
<td>53.57</td>
<td>6.01</td>
<td>105.42</td>
<td>2.19</td>
</tr>
<tr>
<td>VFMHQ1</td>
<td>51.97</td>
<td>5.93</td>
<td>45.74</td>
<td>0.67</td>
</tr>
<tr>
<td>VFMHQ2</td>
<td>54.26</td>
<td>6.26</td>
<td>45.37</td>
<td>0.66</td>
</tr>
<tr>
<td>VFMHQ3</td>
<td>47.83</td>
<td>5.85</td>
<td>45.05</td>
<td>0.66</td>
</tr>
<tr>
<td>VFMHQ4</td>
<td>57.29</td>
<td>6.46</td>
<td>45.58</td>
<td>0.67</td>
</tr>
<tr>
<td>VMFQ</td>
<td>89.31</td>
<td>7.27</td>
<td>140.93</td>
<td>1.00</td>
</tr>
</tbody>
</table>
Table 2.3: A quantitative comparison of the nonlinear interpolators with other sample images for the case of rectangular decimation

<table>
<thead>
<tr>
<th>Method</th>
<th>&quot;Collins&quot;</th>
<th>&quot;Fighter&quot;</th>
</tr>
</thead>
<tbody>
<tr>
<td>MSE</td>
<td>MSE</td>
<td></td>
</tr>
<tr>
<td>Pixel Replication</td>
<td>492.1</td>
<td>1357.7</td>
</tr>
<tr>
<td>Bilinear</td>
<td>152.5</td>
<td>450.8</td>
</tr>
<tr>
<td>Cubic spline</td>
<td>171.6</td>
<td>554.2</td>
</tr>
<tr>
<td>VFMH_{R1}</td>
<td>156.1</td>
<td>449.0</td>
</tr>
<tr>
<td>VFMH_{R2}</td>
<td>162.1</td>
<td>462.8</td>
</tr>
<tr>
<td>VFMH_{R3}</td>
<td>149.9</td>
<td>441.4</td>
</tr>
<tr>
<td>VFMH_{R4}</td>
<td>154.1</td>
<td>451.6</td>
</tr>
<tr>
<td>VFMH_{R5}</td>
<td>174.8</td>
<td>492.6</td>
</tr>
<tr>
<td>VFMH_{R6}</td>
<td>149.8</td>
<td>440.7</td>
</tr>
<tr>
<td>VMF_{R}</td>
<td>172.0</td>
<td>524.5</td>
</tr>
</tbody>
</table>
Figure 2.4: Applications of image/video zooming: (a) image zoom and transfer of circular object from video sequence “A” to video sequence “B” by editing and compositing as envisioned in MPEG-4 and 7, and (b) a video sequence is transmitted at a reduced spatial resolution and scaled up to an increased resolution for playback at the receiver.
Figure 2.5: Frequency-domain illustration of downsampling [16].
Figure 2.6: Frequency-domain illustration of interpolation [16].
Figure 2.7: The magnitude of the Fourier Transform provides a comparison of the different interpolating functions in the frequency domain. The dashed box indicates the ideal lowpass filter response: (a) Nearest neighbor, (b) Bilinear, (c) Cubic B-spline, and (d) Cubic spline.
Figure 2.8: Spatial image interpolation of *Lena* image: (a) Original image of *Lena*, (b) downsampling image of the original (i.e. by a factor of 4 in each direction without prefiltering), (c) Pixel replication, (d) Bilinear technique, (e) Cubic spline, (f) $\text{FMH}_{R6}$, (g) VMF, (h) $\text{VFHM}_{R2}$, (i) $\text{VFHM}_{R4}$, (j) $\text{VFHM}_{R6}$. 
Figure 2.9: Spatial image interpolation of a synthetic image: (a) Original synthetic image, (b) downsampled image of the original (i.e. by a factor of 4 in each direction without prefiltering), (c) Pixel replication, (d) Bi-linear technique, (e) Cubic spline, (f) FMH_{R6}, (g) VMF, (h) VFMH_{R2}, (i) VFMH_{R4}, (j) VFMH_{R6}.
Chapter 3

Statistical Methods for Spatial Image Interpolation

3.1 Introduction

The importance of spatial image interpolation was emphasized in the previous chapter where deterministic, nonlinear filter structures were proposed in order to overcome the limitations associated with conventional linear schemes. In this chapter, the problem of interpolation is approached under the statistical framework of Gibbs random field (GRF) models. The application of GRF models to problems in texture modeling and classification [41], segmentation and restoration of degraded images [42], has been studied quite extensively. Here we follow a similar approach by devising the problem of image interpolation in the form of a restoration problem. Different parametric forms of Gibbs random fields have been applied to this problem [11, 43, 44].

GRF modeling allows the joint probability distribution of the image pixels to be expressed in terms of an energy function which provides a powerful mechanism for modeling spatial continuity. The GRF representation of images is first introduced, followed by the approach used in estimating the parameters of the model. Image regeneration or interpolation is subsequently described using various pixel site replacement algorithms. Finally, the site replacement schemes are implemented via nonlinear filtering process using binary logic-type filters.
3.2 Modeling Images using Gibbs Random Fields

There has been an increasing interest in the use of statistical techniques for modeling and processing of image data. Random field models have recently been applied towards problems in texture modeling and classification, segmentation, and restoration. The reports by Cross and Jain [41], Geman and Geman [45], and Derin and Elliot [42], all make use of Gibbs distributions for characterizing random fields. Here, we represent an image by a Gibbs random field (GRF) model and apply this to the interpolation problem.

An image can be considered as a 2D random field, $X$ over a finite rectangular lattice, $L$ where

$$L = N_1 \times N_2$$

(3.1)

The random field, $X$ is a collection of random variables (RV's), $X_{i,j}$ for all $(i, j) \in L$. The lowercase $x_{i,j}$ is the specific value that the RV, $X_{i,j}$ takes, and $x$ is the collection of all $x_{i,j}$ (i.e. a sample realization). A neighborhood system, $\eta$ of the lattice of points is a collection of subsets of $L$

$$\eta_{i,j} \subset L$$

(3.2)

The neighborhood of pixel $(i, j)$ is such that [46]

1. $(i, j) \notin \eta_{i,j}$

2. If $(k, l) \in \eta_{i,j}$ then $(i, j) \in \eta_{k,l}$ for any $(i, j) \in L$

A hierarchical ordered sequence of neighborhood systems $\eta^1, \eta^2, \eta^3, \ldots$ are such that $\eta^1_{i,j}$ consist of four pixels neighboring $(i, j)$ and $\eta^2_{i,j}$ having eight pixels neighboring $(i, j)$. The neighborhood system $\eta^m$ is called the $m^{th}$ order neighborhood system. The neighborhood structure for $m = 1, 2, \ldots, 6$ is shown in Figure 3.1. Neighborhoods, $\eta^1$ and $\eta^2$ are commonly used in image modeling due to their simple, yet powerful representation capability.

A random field, $X$ is a Markov random field (MRF) with respect to the neighborhood system $\eta$ if

$$P(X = x) > 0$$

(3.3)
Figure 3.1: **Hierarchically arranged neighborhood systems** $\eta^m$ [46].

$$P(X_{i,j} = x_{i,j} \mid X_{k,l} = x_{k,l}, (k,l) \neq (i,j)) = P(\quad)$$

for all $(i,j) \in L$ and $x \in \Omega$, where $\Omega$ is the set of all possible realizations [45, 46]. This MRF definition is characterized by the conditional distribution in (3.4) above (i.e. the local characteristics of the random field). However, there are inherent difficulties with the MRF formulation due to the unavailability of the joint distribution, $X$, and consistency problems concerning the joint distribution [47, 48]. Nevertheless, the Hammersley-Clifford theorem [48] establishes an equivalence between an MRF and a Gibbs random field (GRF). Unlike the MRF characterization, the GRF provides us with a practical way of specifying the random field through potentials, thereby eliminating the modeling and consistency problems [45].

In defining a GRF, we must introduce the notion of cliques associated with the lattice-neighborhood pair $(L, \eta)$. A clique, $c$ of the pair $(L, \eta)$ is such that [46]

1. $c$ consists of a single pixel, or

2. for $(i,j) \notin (k,l)$, $(i,j) \in c$ and $(k,l) \in c$ implies that $(i,j) \in \eta_{k,l}$

The collection of all cliques of $(L, \eta)$ is denoted by $C = C(L, \eta)$. The various cliques associated with $\eta^1$ and $\eta^2$ are shown in Figure 3.2. In this work, we utilize the second order neighborhood, $\eta^2$ with clique pair types only.
Figure 3.2: Neighborhood systems $\eta^1$ and $\eta^2$ and their associated clique types [46].

The joint distribution of a Gibbs random field is given by

$$P(X = x) = \frac{1}{Z} e^{-U(x)/T}$$  \hspace{1cm} (3.5)$$

$$U(x) = \sum_{c \in C} V_c(x)$$ \hspace{1cm} (3.6)$$

where $U$ is the energy function, $V_c$ is the clique potential, $T$ is a temperature parameter, and $Z$ is a normalizing constant. The most general form of the energy function, $U$ is as follows

$$U(x) = \sum V_1(x_{ij}) + \sum V_{2v}(x_{ij}, x_{i,j+1}) + \sum V_{2h}(x_{ij}, x_{ij+1}) +$$
$$\sum V_{2d1}(x_{ij}, x_{i+1,j-1}) + \sum V_{2d2}(x_{ij}, x_{i,j+1}) + \ldots + \sum V_n(x_{ij}, x_{i,j+1}, x_{i+1,j}, \ldots)$$ \hspace{1cm} (3.7)$$

where $V_1$ is the potential associated with single pixels, $V_2$, the potential of clique pairs (i.e. $V_{2v}$ are the vertical cliques, $V_{2h}$ the horizontal cliques, and $V_{2d}$ the diagonal...
cliques), and $V_n$ the potential of $n$ pixel cliques. The summations in equation (3.7) extend over all $(i, j) \in L$. Here, a set of binary valued potential functions are chosen over a second order neighborhood system providing a simple yet powerful representation

$$V_{2v} = \begin{cases} -\beta_1 & |x_{i,j} - x_{i+1,j}| < t_s \\ \beta_1 & \text{otherwise} \end{cases} \quad (3.8)$$

$$V_{2h} = \begin{cases} -\beta_2 & |x_{i,j} - x_{i,j+1}| < t_s \\ \beta_2 & \text{otherwise} \end{cases} \quad (3.9)$$

$$V_{2d_1} = \begin{cases} -\beta_3 & |x_{i,j} - x_{i+1,j-1}| < t_s \\ \beta_3 & \text{otherwise} \end{cases} \quad (3.10)$$

$$V_{2d_2} = \begin{cases} -\beta_4 & |x_{i,j} - x_{i+1,j+1}| < t_s \\ \beta_4 & \text{otherwise} \end{cases} \quad (3.11)$$

where $t_s$ is the pixel threshold value of similarity. Through this energy concept we can effectively represent images using the idea of pixel bonding or clustering, where the parameters, $\beta_i$ for $i = 1, \ldots, 4$ are essentially used to control the size and direction of the clusters within the image. Thus, the estimation of the parameters, $\beta_i$ is the first step in appropriately modeling the image.

### 3.3 Parameter Estimation

The parameters, $\beta_i$ of our GRF can be represented by the parameter vector $\beta = [\beta_1, \beta_2, \beta_3, \beta_4]$. This vector must be estimated to adequately model the random field $X$. Several schemes have been presented in estimating the parameters of a GRF [48, 41, 42]. The coding method in [48, 41], originally proposed by Besag [48], requires the solution of a set of nonlinear equations which are cumbersome and difficult to use reliably. Furthermore, a number of different estimates are obtained from a single realization, and an established method to combine these estimates into one does not exist [42]. In view of these practical difficulties, an alternative approach based on a
histogramming and linear, least-squares estimation technique was devised [42]. In this scheme, a set of probabilities are determined using a single realization of the image. This works well for binary images, however, for images with more than 2 levels such as 256 gray level images, there is insufficient data in a typically sized image for an accurate histogramming. For this purpose, we use a more rigorous iterative parameter estimation technique using a gradient ascent method. In this approach, the objective function $Q(\beta, \tilde{\beta})$ (i.e. the log likelihood ratio) below is maximized through an iterative process where $\beta$ is the parameter vector estimated from the previous iteration, and $\tilde{\beta}$ the parameter vector yet to be estimated [49]

$$Q = \log \left[ \frac{P(X | \tilde{\beta})}{P(X | \beta)} \right]$$

(3.12)

Each application of this expression increases the probability of observing $X$, until a point is reached where the new estimate is within a threshold of the previous value. It is shown that iteratively maximizing $Q$, gives the same parameter estimate as the direct maximization of the log likelihood function [49].

The gradient of $Q(\beta, \tilde{\beta})$ with respect to the parameter vector $\tilde{\beta}$ leads to the following parameter update equation [49]

$$\tilde{\beta}^{(k+1)} = \tilde{\beta}^{(k)} + \lambda^{(k)} \nabla_{\beta^{(k)}} Q[\beta^{(k)}, \tilde{\beta}^{(k)}]$$

(3.13)

where $\lambda^{(k)}$ is the step size for the update of $\tilde{\beta}$ at the $k^{th}$ iteration, and $\nabla$ is the gradient function. Thus, for our second order neighborhood, we have the four equations below

$$\tilde{\beta}_1^{(k+1)} = \tilde{\beta}_1^{(k)} + \lambda^{(k)} \frac{\partial Q[\beta_1^{(k)}, \tilde{\beta}_1^{(k)}]}{\partial \tilde{\beta}_1^{(k)}}$$

(3.14)

$$\tilde{\beta}_2^{(k+1)} = \tilde{\beta}_2^{(k)} + \lambda^{(k)} \frac{\partial Q[\beta_2^{(k)}, \tilde{\beta}_2^{(k)}]}{\partial \tilde{\beta}_2^{(k)}}$$

(3.15)

$$\tilde{\beta}_3^{(k+1)} = \tilde{\beta}_3^{(k)} + \lambda^{(k)} \frac{\partial Q[\beta_3^{(k)}, \tilde{\beta}_3^{(k)}]}{\partial \tilde{\beta}_3^{(k)}}$$

(3.16)
The gradient of the objective function for $\beta_i$ can be computed as

$$\beta_i^{(k+1)} = \beta_i^{(k)} + \lambda^{(k)} \frac{\partial Q[\beta_i^{(k)}, \beta_i^{(k)}]}{\partial \beta_i^{(k)}}$$

where $I(x)$ is a modified indicator function

$$\frac{\partial Q[\beta_i^{(k)}, \beta_i^{(k)}]}{\partial \beta_i^{(k)}} = \frac{1}{(N_1)(N_2 - 1)} \left[ \sum_{i=1}^{N_1} \sum_{j=1}^{N_2-1} I_i(x_i, x_{i+j}) - \sum_{i=1}^{N_1} \sum_{j=1}^{N_2-1} I_i(x_i, x_{i+j}+1) \right]$$

Expressions similar to (3.18) can be formed for the parameters, $\beta_i$ for $i = 2, \ldots, 4$. The iterative scheme described above admits a unique estimate of $\tilde{\beta}$ for our selected energy function [49].

### 3.4 Image Interpolation

In image regeneration, or interpolation, we begin with a downsamplesd image along with the estimated parameter set of the Gibbs distribution. The first step in determining the upsampled image requires the insertion of randomly chosen pixel values in every other row and every other column (EORC) of the downsamplesd image. In this chapter, we only consider the rectangular sampling lattice. We also note that pixel insertion does not necessarily require that random values be chosen. The interpolation problem consists of determining the unknown pixel values (i.e. randomly inserted pixels) so that they are as close as possible to the original values. This can also be seen as a restoration problem where the pixels in every other row and column are known to be degraded. Ideally, we would like to determine the maximum a posteriori (MAP) estimate of the original image, given the randomly interlaced image and the
Gibbs parameter set, that is

\[ \hat{x} = \text{MAP} \left[ P(X \mid Y, \beta) \right] \]  

(3.20)

where \( \hat{x} \) is the interpolated image, \( X \) is the original image, and \( Y \) is the randomly interlaced image. However, MAP estimation presents a formidable computational problem. The number of possible intensity images is \( 256^{(N_1/2)(N_2/2)} \) for a 256 gray-level image of dimensions \( N_1 \times N_2 \) which rules out any direct search. Consequently, we must generate a sequence of images which ideally converge to the MAP estimate through an evolution of local changes based upon the pixels in the neighborhood.

Here, we examine three iterative site-replacement algorithms where the image gradually evolves by discrete changes at individual pixels: (i) the Gibbs Sampler (GS), (ii) Simulated Annealing (SA), and (iii) the Iterative Conditional Mode (ICM) method.

The Gibbs Sampler generates realizations from a given random field by a pixel visiting mechanism \[45, 50\]. The algorithm is briefly described as follows. We begin by proceeding in a raster scan fashion over the image, and visiting each pixel site sequentially (in our case, only the pixels which must be interpolated are visited). At each site visit, the pixel is replaced with a randomly selected pixel value. The state of the system prior to replacement is \( X(t) \), and afterwards the new state becomes \( X(t+1) \). We then determine whether the probability of the new state has increased in the local neighborhood. The ratio of the two probabilities, \( q \) is as follows

\[ q = \frac{P[X(t+1)]}{P[X(t)]} = \frac{e^{-U[X(t+1)]}}{e^{-U[X(t)]}} \]  

(3.21)

where the temperature, \( T \) in equation (3.5) is set to a constant value of one. Thus, the ratio simplifies to

\[ q = e^{-\Delta U} \]  

(3.22)

\[ \Delta U = U[X(t+1)] - U[X(t)] \]  

(3.23)

where \( \Delta U \) is the energy change which is computed using the previously estimated model parameters, \( \beta_i \). If \( q > 1 \), then the pixel is replaced, whereas if \( q \leq 1 \), then
the transition is made with probability $q$. If $s_1, s_2, \ldots, s_n$ is the sequence in which sites are visited for replacement, then for every starting configuration $x(t = 0) \in \Omega$ and every $x(t = n) \in \Omega$ [45]

$$
\lim_{n \to \infty} P [X(t = n) = x(t = n) \mid X(t = 0) = x(t = 0)] = X
$$

Thus, the original image, $X$ is approached in the limit through the GS site replacement algorithm. This stochastic relaxation process permits changes which increase the energy (i.e. lower probability) and, thereby, avoids undesirable local maxima typical of deterministic algorithms.

To speed up the process of the Gibbs Sampler, the technique of Simulated Annealing is employed [51]. In this approach, the temperature parameter, $T$ is gradually lowered simulating an annealing procedure of chemical systems. At low temperatures, the local distributions concentrate on states that increase the probability function, whereas at high temperatures, the distribution is essentially uniform [45]. Local maxima are avoided at the initial high temperatures where many of the stochastic changes will actually decrease the probability function. As the relaxation proceeds, the temperature is gradually lowered and the process iteratively improves. The lowering of the temperature in the annealing theorem guarantees convergence to the global maxima of the posterior distribution (i.e. MAP estimate) [45]. An inverse exponential function is utilized here as the annealing schedule.

Simulated annealing, although it can guarantee the convergence to the global optimum, is computationally demanding and may be impractical in many applications. The iterative conditional mode (ICM) is an alternative optimization technique that can reduce the computational load [48, 52]. In the ICM method, at each site visit, the energy is computed for all possible pixel values, and the value with the highest probability is selected. Thus, a successive maximization of the probability function is carried out at each individual pixel. This does not necessarily converge to the MAP estimate, but the local maximization process yields a good approximation at a reduced complexity [52]. The iterative process can be repeated until no change
occurs in the estimation, or the change is below a preset threshold. We apply the ICM method to the interpolation problem, and then relate this process to a nonlinear filtering scheme where binary logic-type filters are used. Finally, the iterative ICM technique is approximated by implementing the proposed filters over three passes of the image.

In viewing the ICM process as a nonlinear filtering scheme, we first examine the simple binary valued energy function over a second order neighborhood in equations (3.8-3.11). At any pixel site only the four surrounding pixels have any effect on the conditional probability function. Clearly, the pixel \( x_{i,j} \) chosen must result in the smallest value for the sum \( V = V_{2v} + V_{2d} + V_{2d} + V_{2d_{2}} \) in order to have the highest probability of occurrence, as in ICM. If for example, the pixel \( x_{i,j} \) falls within \( t_{s} \) of the pixel \( x_{i+1,j} \) then \(-\beta_{1}\) is contributed towards the sum. If at the same time this value also is within \( t_{s} \) of \( x_{i,j+1} \) then \(-\beta_{2}\) is added towards the sum and so forth. Not all pixel values in the range of gray levels will produce a different value for the sum. Thus, only pixels that produce different sums need to be checked. These pixels can be determined as follows. Each of the four neighboring pixels \( x_{i+1,j}, x_{i,j+1}, x_{i+1,j-1}, x_{i+1,j+1} \) has an associated region of similarity \( R_{1}, R_{2}, R_{3}, R_{4} \), respectively. The region or the set \( R_{1} \) consists of the pixels in the range \([x_{i+1,j} - (t_{s} - 1)] \) to \([x_{i+1,j} + (t_{s} - 1)] \) and \( R'_{1} \), the complement of \( R_{1} \), consists of all other pixel values in the range of gray level values. The set \( R_{1} \) has an associated weight attached to it which is \(-\beta_{1}\) and \( R'_{1} \) is \( \beta_{1} \). Regions \( R_{2}, R_{3}, R_{4} \) are formed in a similar fashion. The sum \( V \) is computed only for different overlapping regions. If there is an overlap of all four regions then \( V = -\beta_{1} - \beta_{2} - \beta_{3} - \beta_{4} \) and clearly this results in the smallest possible value for \( V \).

However, if there is no intersection of these four regions then the sums must be computed for different combinations of three overlapping regions (i.e. \( R_{2} \cap R_{3} \cap R_{4} \)), two overlapping regions, and single regions. In total, 15 combinations must be checked. The smallest sum is chosen. This reduces the number of checks if all 256 gray level values must be tested. Furthermore, these set operations can be implemented via binary logic-type operations shown in the next section.
3.4.1 Nonlinear Filtering Using Binary Logic-Type Operations

The set operations indicated in the previous section can be performed using binary logic-type filters as will be described below. Any region $R_i$ can be stored as a bit pattern in a 256 bit-wide register where the least significant bit represents the pixel value 0 and the most significant the value 255. The bits which correspond to the pixel values in the region are marked with 1's and all other bits are set to 0's. Therefore, a specific bit pattern is generated for a particular region. If there are 256 gray levels, then there are 256 possible bit patterns (i.e. 256 possible regions) for any chosen threshold $t_s$. These patterns can be stored in the form of a look-up table (LUT) which can consist of 256 different registers each 256 bits wide. In hardware this can be 64K of non-volatile memory. The set operations can now be implemented by bitwise logical AND operations. Thus, for each of the four neighborhood pixel values (i.e. $x_{i+1,j}$, ...) its corresponding bit pattern can be found from the LUT. Bitwise logical AND operations are performed in place of the set operations shown earlier. The output value of the AND operation is taken and a bitwise OR operation is subsequently performed. If the result is 1 then there is region overlap otherwise a 0 is encountered which indicates no overlap. For each combination of overlapping regions there is an associated sum of the parameter values (i.e. for $R_1 \cap R_2 \cap R_3$, $V = -\beta_1 - \beta_2 - \beta_3 + \beta_4$). These sums can be precomputed and stored for each combination of overlapping regions and stored in increasing order for a particular parameter set. Thus, at the interpolation stage the bitwise AND and OR operations are performed in this order, so that probabilities of pixel values are checked in decreasing order. If a 1 is found in the output then the current pixel replaces the previous one, and the algorithm proceeds to the next site for pixel replacement. These filtering operations just described duplicate the ICM process in a much more efficient manner. Further simplification is possible by eliminating the iterative procedure and reducing the number of operations per pixel. A 3-pass filter that implements the logic-type operations described above can be used to simplify and approximate the ICM method. The three stage filtering
process operating on the subset of pixels of the upsampled image is described below

\[
\begin{align*}
M & N_2 & M & N_2 & M \\
N_2 & N_1 & N_2 & N_1 & N_2 \\
M & N_2 & M & N_2 & M
\end{align*}
\]

where \( N_1 \) and \( N_2 \) are the pixels to be interpolated, and \( M \) are the known pixels from the downsampled image. In the first pass of the image, the pixels \( N_1 \) are determined using each pixel's south-east and south-west neighbors. In the second pass, the pixels \( N_2 \) are found using each ones' east and south neighbors. In the final pass all of the pixels \( N_1, N_2 \) are recomputed from each ones' corresponding four neighbors (i.e. east, south-east, south, south-west). In the first two passes, only one logical AND and one OR operation are required per pixel and in the third pass only 45 operations are required per pixel. This greatly reduces the computational burden of the ICM method and yields visually similar results as found in the experiments.

### 3.5 Experimental Results

The three site replacement techniques and the 3-pass filter described above, were applied to two gray-level images: (i) a textured region of the Mandrill image, and (ii) a section of Lena which consists of both, smooth areas and edges. Smaller sections of the original images were selected due to the demanding computational complexity of the parameter estimation stage. In Figure 3.3, the results of the Mandrill image are shown. In part (a), the original is illustrated, and in part (b), the selected \( 64 \times 64 \) textured region (i.e. whiskers) is shown. The image in (b) was downsamped by two in each direction (i.e. rectangular sampling) and subsequently upsampled by the various techniques. The two images in parts (c) and (d) of the figure demonstrate the results of the conventional linear schemes. As expected, the pixel replication method in part (c) suffers from artifacts in the form of blockiness, while the bilinear scheme of (d) is degraded by blurring and aliasing. The results of the statistical methods are shown in the final four parts of the figure. The parameters were estimated for each \( 8 \times 8 \)
block over the $64 \times 64$ textured region. This block size is small enough to adapt to smaller textured regions yet large enough to keep the computational complexity and the number of parameters relatively low. In part (e) of Figure 3.3, the site replacement scheme of the Gibbs Sampler is illustrated. The results were obtained by 200 iterations of pixel changes within the image (i.e. each iteration corresponds to one full raster scan of the image). The GS technique attempts to reproduce the original section, however, we notice distinct impulsive noise-like pixels in the figure. This is due to the fact that the temperature parameter is set to a constant value of one (i.e. higher temperature values). As a result of this, there is a loose coupling between neighboring pixels which allows some purely random changes. The technique of Simulated Annealing is used to improve upon the GS method, and the performance of this scheme is shown in part (f). The temperature was lowered in an inverse exponential fashion (with $T = 1$ initially) according to $T = 1/2^{\text{int}(k/5)}$ where \text{int} retains the integer component, and $k$ is the iteration number. In other words, the temperature is reduced for every five iterations of pixel changes. The results are shown for 50 iterations of the image. We note, that the SA technique removes some of the impulsive artifacts of the GS method and yields a very reasonable interpolated image. In part (g) of the figure, the output image from the ICM technique is shown. Once again, the performance of this scheme is quite reasonable and close to the SA method. Finally, in part (h), the approximating 3-pass nonlinear filtering results are illustrated. The non-iterative filtering scheme does a good job in approximating the ICM method at a much reduced computational complexity. The latter three parts of Figure 3.3 (i.e. f, g, and h) reproduce reasonable approximations of the original in part (b), and are free from the artifacts present in the conventional linear techniques. However, these iterative schemes have a significant increase in computational complexity. An additional performance improvement can be realized by increasing the number of iterations carried out in the iterative schemes. The 3-pass filter reduces the complexity, but still requires an additional 256 bytes for the model parameters. Quantitative measures are not provided as the small size of the image will not yield sufficiently meaningful results.

Figure 3.4 illustrates the results for a small section of the \textit{Lena} image which
consists of smooth areas as well as edges. The impulsive noise-like pixels are readily apparent in the GS method, while the SA, and the ICM methods are much improved. Nevertheless, the artifacts appear to be more severe in the geometrical-type figures (i.e. smooth areas as well as edges) such as Lena as compared to the texture-type images as in the Mandrill image. In future work, a line process [45] which contains edge information could also be used to improve the results of the geometrical images. Finally, the 3-pass filter in part (h) yields a reasonable approximation to the original with some loss of detail. Thus, the statistical methods perform quite satisfactorily for both, smooth and textured regions as demonstrated by the experimental results, but appear to be more suitable for the textured images.

3.6 Conclusions

Statistical techniques using Gibbs random fields models were used to obtain an interpolated image from its downsampled version. Three iterative techniques, that of the Gibbs Sampler, Simulated Annealing, and Iterative Conditional Mode, as well as a non-iterative 3-pass filter based on binary logic-type operations were implemented. It was found experimentally that the latter produced visually similar results to the former three methods which are both computationally expensive. The 3-pass filter also permits a simple hardware solution for real-time applications. Experimental results also indicated that the statistical techniques did not suffer from the artifacts present in the conventional linear schemes. The proposed techniques appeared to be more suitable for textured images, however, the use of edge information in the form of a line process could be used to improve the performance for the geometrical-type images.
Figure 3.3: (a) Original Mandrill image, (b) a textured region of the original image, (c) pixel replication, (d) bilinear interpolation, (e) Gibbs Sampler, (f) Simulated Annealing, (g) Iterative Conditional Mode, and (h) the 3-pass nonlinear filter.
Figure 3.4: (a) Original Lena image, (b) a smaller section of the original image, (c) pixel replication, (d), bilinear interpolation, (e) Gibbs Sampler, (f) Simulated Annealing, (g) Iterative Conditional Mode, and (h) the 3-pass nonlinear filter.
Chapter 4

Facial Image Segmentation

4.1 Motivation and Related Work

The computer recognition of human faces is currently an active area of research in computer vision [53, 54, 55, 56]. The task of recognizing human faces is essentially a two step process: 1) The detection and automatic location of the human face, and 2) the automatic identification of the face based on the extracted features [57]. Most of the research to date has been directed towards the latter identification phase, with less emphasis being placed on the initial localization stage. However, the first step is critical to the success of the second and the overall recognition system. Thus, the importance of obtaining an accurate localization of the face is clear and vital in numerous applications including human recognition for security purposes, human-computer interfaces, and more recently, for object-based video coding, video databases, and video on demand. Nevertheless, determining the location of a face of unknown size, in a scene with a complex or moving background still remains a difficult problem that is relatively unexplored.

Several techniques based on shape and motion information have been proposed recently for the automatic location of the facial region [58, 59, 57]. The former two are related to video coding applications while the latter is part of a facial recognition
system. The shape-based approach in [58] models the contours of the face as an ellipse. The location of the facial region is determined by performing an ellipse fitting task to a thresholded binary edge image. In [59], a generic 3-D face model is adapted to the extracted facial outline from a videophone type scene for the case where only one person is talking against a stationary background. In this application, a hierarchical localization scheme is utilized to isolate the facial area. The technique is based on the shape of the extracted head-and-shoulders silhouette which is obtained using the thresholded frame differences. Finally, in [57], a motion detection algorithm is used to segment the facial area from a complex background. The proposed method locates the facial region by assuming that the object having the greatest motion in the video sequence is the face to be detected. This assumption however, may limit the success of the approach in applications with non-stationary backgrounds, (i.e. mobile videophones) and/or other moving objects in the scene. The authors also acknowledge potential problems caused by noise or other objects moving in the background and also suggest a modification in their technique to better handle the case of tilted or turned faces.

In this chapter, we focus on the automatic location and tracking of the facial region of a head-and-shoulders videophone-type sequence using color and shape information. The method we present utilizes the skin-tone distribution of the histograms in the HSV color space to initially extract the facial region. The segmentation results are then refined using a series of post-processing operations which include median filtering, region filling and removal, and morphological opening and closing operations. A series of fuzzy membership functions are finally used to correctly classify and retain the facial area in the case of additional falsely included regions. This situation may occur when other objects in the scene have colors similar to those of skin tone regions. The aggregation of these features within the framework of a knowledge-based decision system provides the mechanism of selecting the facial area from the set of candidate regions. The feature vector obtained from this last step can be used to augment a further feature extraction stage which can also support future “content-based” functionalities. Our approach is found to be robust with regards to facial
shape, size, skin color, orientation, motion, and lighting conditions. Furthermore, it can be implemented at a relatively low computational complexity due to the binary nature of the operations performed.

4.2 Color Image Segmentation

Color is a key feature used to understand and recollect the contents within a scene. It is found to be a highly reliable attribute for image retrieval as it is generally invariant to translation, rotation, and scale changes [60]. In our approach we use color as the primary tool in detecting and locating the facial region in a scene with a complex or moving background. The segmentation of a color image is the process of classifying the pixels within the image into a set of clusters with a uniform color characteristic. The objective in our approach is to detect and isolate the color clusters that correspond to the skin areas of the facial region. However, the shape or distribution of the clusters that form depend on the chosen color space [61]. Therefore, the most advantageous color space must first be selected in order to obtain the most effective results in the segmentation process.

4.2.1 Color Coordinate Systems

Several color coordinate systems have come into existence for establishing a numerical description of color. The representation of color is based on the classical three-color theory whereby any color can be reproduced by mixing an appropriate set of three primary colors [62]. In this way, the numerical representation of a particular color can be specified by its three component vector within the 3D color coordinate system. The set of all colors form a vector space called the color space or color model.

Color information is commonly represented in the widely used RGB (Red, Green, Blue) Cartesian coordinate system. This basis is hardware-oriented and is suitable for acquisition or display devices but not particularly applicable in describing the perception of colors. In this coordinate space, the RGB primaries are additive where the individual contributions of each primary are added to form the overall result.
The YIQ (Y is the Luminance, and I,Q the Chrominance components) and CMYK color models are also hardware-based systems and are utilized for different application purposes. The former is used in color television broadcasting, and is a recoding of the RGB components for transmission efficiency and downward compatibility with the earlier monochrome TV standards. The CMYK color space on the other hand, is important in dealing with printing devices where subtractive primaries are relevant. Colors are specified in this latter model by what is removed or subtracted from white light, rather than by what is added to black.

The need to formulate a simple, yet accurate perceptual color distance prompted the development of a perceptually uniform color space [63]. The Commission Internationale De L'Eclairage (CIE) standardized the perceptually uniform L* u*v* and L* a*b* coordinate systems which are derived by a nonlinear transformation of the RGB values. These color models define an uniform metric-space representation of color so that a perceptual color difference is represented by the Euclidean distance. The L* a*b* cube-root color coordinate system was essentially developed to provide a quantitative expression for the Munsell system of color classification [64]. The following transformation equations can be used to convert a set of RGB vector values to the L* a*b* space.

\[
\begin{bmatrix}
X \\
Y \\
Z
\end{bmatrix} = \begin{bmatrix}
0.490 & 0.310 & 0.200 \\
0.177 & 0.813 & 0.011 \\
0.000 & 0.010 & 0.990
\end{bmatrix}
\begin{bmatrix}
R \\
G \\
B
\end{bmatrix}
\]

(4.1)

\[
L^* = 25 \left( \frac{100Y}{Y_0} \right)^{\frac{1}{3}} - 16
\]

(4.2)

\[
a^* = 500 \left[ \left( \frac{X}{X_0} \right)^{\frac{1}{3}} - \left( \frac{Y}{Y_0} \right)^{\frac{1}{3}} \right]
\]

(4.3)

\[
b^* = 200 \left[ \left( \frac{Y}{Y_0} \right)^{\frac{1}{3}} - \left( \frac{Z}{Z_0} \right)^{\frac{1}{3}} \right]
\]

(4.4)
where the constraint $1 \leq 100Y \leq 100$ must be satisfied which is indeed the case for most practical purposes [65]. The intermediate values $[XYZ]^T$ are the CIE XYZ tristimulus values, and the $[X_0 Y_0 Z_0]^T$ triplet is the reference white. In the equations above, $L^*$ is correlated with brightness, $a^*$ with the red-green content, and $b^*$ with the yellow-blue content within the image. A similar set of nonlinear expressions can be found for the $L^*u^*v^*$ coordinate system. The computational complexity of the cube-root expressions above, however, may render the perceptually uniform spaces unsuitable for real-time applications. Comprehensive descriptions of the numerous color coordinate systems can be found in [65, 66, 67, 68] along with their appropriate transformation equations.

The HSV (Hue, Saturation, Value), and the TekHVC (Hue, Value, Chroma) color models belong to a group of Hue-oriented color coordinate systems which correspond more closely to the human perception of color. These user-oriented color spaces are based on the intuitive appeal of the artist’s tint, shade, and tone. The proprietary, TekHVC model was developed by Tektronix as a modification of the CIE $L^*u^*v^*$ perceptually uniform color space described earlier. The HSV coordinate system, originally proposed by Smith [69] is cylindrical and is conveniently represented by the hexcone model shown in Figure 4.1. The Hue (H) is measured by the angle around the vertical axis and has a range of values between 0 and 360 degrees beginning with Red at 0°. It gives us a measure of the spectral composition of a color. The Saturation (S) is a ratio that ranges from 0 (i.e. on the V axis), extending radially outwards to a maximum value of 1 on the triangular sides of the hexcone. This component refers to the proportion of pure light of the dominant wavelength and indicates how far a color is from a gray of equal brightness. The Value (V) also ranges between 0 and 1 and is a measure of the relative brightness. At the origin, $V=0$ and this point corresponds to “black”. At this particular value, both, H and S are undefined and meaningless. As we traverse upwards along the V axis we perceive different shades of gray until the endpoint is reached (where $V=1$ and $S=0$) which is considered to be “white”. At any point along the V axis the Saturation component is zero and the Hue is undefined. This singularity occurs whenever $R=G=B$. The set of equations below can be used
to transform a point in the RGB coordinate system to the appropriate value in the HSV space

\[ H = H_1 \quad , \quad \text{if } B \leq G \]  

\[ H = 360^\circ - H_1 \quad , \quad \text{if } B > G \]  

\[ S = \frac{\text{Max}(R,G,B) - \text{Min}(R,G,B)}{\text{Max}(R,G,B)} \]  

\[ V = \frac{\text{Max}(R,G,B)}{255} \]  

In the expressions above, the \( \text{Max} \), and \( \text{Min} \) operators select the maximum and
minimum values of the operand, respectively, and R, G, and B range between 0 and 255. A fast algorithm used here to convert the set of RGB values to the HSV color space is provided in [66].

4.2.2 Color Space Selection

The segmentation of the skin areas within an image is most effective when a suitable color space is selected for the task, as mentioned earlier. This is the case when the skin clusters are compact, distinct, and easy to extract from the color coordinate system. The complexity of the algorithm must also be low to facilitate real-time applications.

In [57], the color distribution of a facial image was examined using four different color coordinate systems, which included the RGB space, HSI (similar to HSV), CIE L*u*v*, and the Karhunen-Loeve transformation. The HSI color space was found to be the most suitable as it produced clusters that were clearly separated, allowing them to be detected and readily extracted. The other three spaces showed ambiguity in the partitioning of these clusters. We have found similar results by examining the RGB, HSV, and the L*a*b* color spaces. These three coordinate systems cover the different color space groups (hardware-based, perceptually uniform, and Hue-oriented) and are frequently selected color models for testing the performance of many proposed color image segmentation algorithms.

The data from two different skin-colored regions, as well as the lip area from a different set of images were manually extracted and plotted in each of the aforementioned coordinate systems in order to observe the clusters formed. The results obtained from the RGB space are shown in Figure 4.2. From the figure, we note that the skin clusters are positioned relatively close to one another, however, the individual clusters are not compact. Each forms a diagonal, elongated shape that makes the extraction process difficult. In Figure 4.3, the skin and lip clusters are displayed in the L*a*b* color space. In this case, the individual clusters are more compact but are spaced quite a distance apart. In fact, the Euclidean distance from skin cluster #1 to the lip cluster is roughly equivalent to that from skin cluster #1 to #2. Thus, the skin clusters do not have a global compactness which once again
Figure 4.2: **Skin and Lip Clusters in the RGB color space model.**

makes them difficult to isolate and extract. The L*a*b* space is also computationally expensive due to the cube-root expressions in the transformation equations. Finally, in Figure 4.4, the Hue component of the skin and lip clusters from the HSV space are shown. The graph illustrates that the spectral composition of the skin and lip areas are distinct, and compact. Skin clusters #1 and #2 are contained between the Hue range of 10 and 40° while the lip region lies at a mean Hue value of about 2° (i.e. close to the Red Hue value at 0°). Thus, the skin clusters are well partitioned allowing the segmentation to be performed by a thresholding scheme in the Hue axis.
Figure 4.3: **Skin and Lip Clusters in the L* a* b* color space model.**

rather than a more expensive multidimensional clustering technique. The HSV model is also advantageous in that the mean Hue of the skin values can give us an indication of the skin tone of the facial region in the image. Average Hue values closer towards $0^\circ$ contain a greater amount of reddish spectral composition while those towards $60^\circ$ contain greater yellowish spectral content. This can be useful for content-based storage and retrieval for MPEG 4 and 7 applications as well as multimedia databases. On the contrary, central cluster values in the other coordinate systems, (i.e. $[R_c \ G_c \ B_c]^T$ or $[L_c^* \ a_c^* \ b_c^*]^T$) do not provide the same meaningful description to a human observer.
Having defined the selected HSV color space, we must subsequently devise a technique to determine and extract the color clusters that correspond to the facial skin regions. This requires an understanding of where these clusters form in the space just outlined in the previous section. We examine the distribution of these clusters next.

### 4.2.3 Skin-tone Distribution

Human skin is composed of several layers of tissue which consist essentially of blood cells, and a yellow pigment called melanin [70]. The appearance of the skin is affected by a number of factors which include the degree of pigmentation (varies amongst individuals and different races), the concentration of blood, and the incident light
source. The combination of all of these factors give rise to a variation in skin color which spans over the range of red, yellow, and brownish-black. Nevertheless, this corresponds to a restricted range of Hue values as will be shown below. In [71], a Hue range that is representative of skin regions has also been proposed.

A large sample of head-and-shoulders type images were collected to observe the distribution of skin colors in the HSV color space. The test images contained several MPEG 4 test sequences, as well as numerous still images obtained from the Internet. The test set consisted of facial images from different races, in order to model a wide range of skin colors. These included Caucasian, Asian, and African-American skin-types. The following scheme was used to generate the histograms for the H, S, and V components of each category. The facial skin region was manually selected in each sample image, and the H, S, and V values were determined for each pixel within this area. The histograms were subsequently formed by compiling the results from all of the images within each category. The normalized histograms obtained from this procedure are shown in Figure 4.5.

It is clear that in all three categories the Hue component consists of a limited range of values. The Hue values of Caucasian and Asian samples fall predominantly between $0^\circ$ (Red) and $60^\circ$ (Yellow) while those of African-American are shifted closer towards $0^\circ$ with a small portion of the distribution in the Red-Magenta Hue sector. One may also note that the Hue values between $180^\circ$ and $360^\circ$ can be represented by their equivalent negative values (i.e. $340^\circ = -20^\circ$). In the figures, the Saturation component ranges from about 10 to 100% in all cases, with the majority falling in the 20-60% range. This suggests that the skin colors for all races are somewhat saturated but not deeply saturated. Finally, we see in Figure 4.5 that the Value or brightness component for both, Caucasian and Asian distributions ranges from approximately 40% to the maximum value of 100%. The Asian test images are shifted even more so towards the maximum value of V (i.e. top of the hexcone model) signifying a high level of brightness in the facial skin regions of these samples. The African-American test set on the other hand, has a wider Value range but is shifted towards lower values. The mean, $m$, and standard deviation, $\sigma$ (both given in degrees), of the three Hue
distributions are conveniently summarized in Table 4.1.

Table 4.1: Statistics of the Hue distribution categorized by race.

<table>
<thead>
<tr>
<th></th>
<th>Caucasian</th>
<th>African-American</th>
<th>Asian</th>
</tr>
</thead>
<tbody>
<tr>
<td>$m$ (°)</td>
<td>25.3</td>
<td>28.9</td>
<td>8.6</td>
</tr>
<tr>
<td>$\sigma$ (°)</td>
<td>6.8</td>
<td>8.2</td>
<td>5.1</td>
</tr>
</tbody>
</table>

The tabulated values indicate that the Asian test samples have the highest mean value of the three distributions, $m = 28.9°$ (i.e. greater shift towards Yellow) with the lowest standard deviation, $\sigma$. The Caucasian sample set has similar statistics with a slightly smaller mean value, $m = 25.3°$ and a slightly larger value of $\sigma$. The African-American distribution has the smallest mean value of the three, $m = 8.6°$ (shift towards Red) and the largest standard deviation. The large value in $\sigma$ can be attributed to the variation in skin colors within the African-American sample set.

Having obtained the distribution for this wide range of skin colors, we must devise an appropriate scheme to segment the facial skin area in any given image. We outline the proposed technique below.

4.2.4 Extraction of Skin-tone Regions

The basis of segmenting an image by color lies in the extraction of a set of regions that satisfy some homogeneity criterion using the spectral components of the image. The approach in any technique depends on the way these regions to be extracted are defined and formed. Four fundamental approaches can be identified and are categorized as follows: 1) Pixel-based techniques, 2) Area-based methods, 3) Edge-based schemes, and 4) Physics-based vision models. In the first of these techniques, the regions to be segmented are determined by operating directly in the color space domain. The set of pixels that form each region are determined by a class membership function which is defined in the selected color space. Histogram-based techniques, clustering, and fuzzy clustering methods all fall into this first category. In the area-based schemes of the second category, the regions of uniformity are determined by operating spatially
in the image domain. Region growing, and split and merge algorithms belong to this class of techniques. In edge-based segmentation, a color contour is created by connecting a set of edge pixels determined by various color edge detectors. Finally, the fourth category belongs to a relatively new class of computer vision methods which employ physical models to partition the image. The aim in this latter approach is to segment the image at the object boundaries rather than the edges of highlights or shadows of the image. An extensive survey of the various techniques can be found in [72].

The method we propose here falls into the first of the four categories described above. The objective in pixel-based segmentation techniques is to partition or divide the color space rather than segment the spatial domain of the image. In histogram-based approaches this partitioning can be accomplished by determining the significant peaks and valleys of the computed histograms and setting the thresholds accordingly. A variety of multi histogram-based thresholding schemes have been suggested to divide multichannel data as in color images [73, 74]. Alternatively, the color space can be divided by using a technique known as clustering [61, 75]. In this scheme, the partitioning is a function of the input vectors (i.e. vector values of the color pixels) and is based on a criterion of optimality such as the least sum of squares. This is closely related to the vector quantization problem of mapping the set of input vectors to a finite number of weight vectors which form the Voronoi tessellation. The computational complexity of these latter techniques can become quite demanding. Either of the two approaches just described can be utilized as general purpose segmentation schemes. However, a scene that consists of an unknown number of homogeneous regions or objects is, in general, very difficult to segment. In many cases, the techniques involve some human interaction in which certain thresholds are manually selected or, where assumptions are made regarding the number of distinct regions or clusters in the scene. In our particular application, we utilize the apriori knowledge of the skin-tone distributions found previously to identify and extract the facial skin regions. A polyhedron is defined in the HSV hexcone model which contains the skin-colored clusters. The proper selection of this polyhedron is the key to obtaining successful
The Hue component is the most significant feature in defining the desired polyhedron. The histograms of Figure 4.5 indicate that the Hue values can be represented by a limited range from $340^\circ - 360^\circ$ (Magenta-Red) and $0^\circ - 50^\circ$ (Red-Yellow) for all skin types. This range is very effective in extracting skin colored regions under higher levels of illumination and sufficiently saturated colors. However, the Hue can be unreliable when the following two conditions arise: 1) When the level of brightness (i.e. Value) in the scene is low, or 2) when the regions under consideration have low Saturation values. The first condition can occur in areas of the image where there are shadows or, generally, under low lighting levels. In the second case, low values of Saturation correspond to achromatic regions. As mentioned previously, Saturation values of zero lie on the V axis in the hexcone model and appear as gray areas. Many objects, by nature, are achromatic (i.e. white clouds, gray asphalt roads, etc.), however, shadows or conditions of non-uniform illumination (i.e. specular reflection) can cause chromatic regions such as skin areas to appear achromatic. Thus, we must define thresholds for the Value, and Saturation components where the Hue attribute is reliable. Incidentally, this will also define the desired polyhedron.

The HSV hexcone model of Figure 4.1 and the distributions of Figure 4.5 were used in the threshold selection process. A lower bound threshold of $T_{\text{val}} = 35\%$ was chosen for the Value component. Pixels with values less than $T_{\text{val}}$ were not considered in the segmentation process as the Hue becomes unreliable for values below this threshold. This can be seen visually by observing the hexcone model as the Value component is varied. Figure 4.6 illustrates four different cases: 1) When the brightness value is at its maximum, $V = 100\%$, 2) at $V = 63\%$, 3) at the threshold value, where $V = T_{\text{val}} = 35\%$, and 4) below the threshold value at $V = 20\%$.

The figure gives us an indication of the discriminatory power of the Hue component at four different slices (i.e. hexagons) of the hexcone model. Radial supersets of the hexagons are shown in the figure for the sake of simplicity. The effectiveness of the Hue is evident in parts a), and b) where the Value is at its maximum, and at $V = 63\%$, respectively. Part d) clearly illustrates that the Hue is meaningless when
the brightness in the scene is low. On the other hand, the threshold value of $V = 35\%$ in part c), is a break point where the Hue component starts to become ineffective. Experimental results also indicated that the selection of a lower threshold led to erroneously detected regions. The importance of intensity information for color image segmentation has also been emphasized in [61, 76, 77].

A Saturation threshold, $T_{sat}$, is also very important in obtaining reliable segmentation results. We have found that the Hue is reliable when the Saturation is greater than 20% and meaningless when it is less than 10%. Similar results have been determined in [76]. The sector between 0 and 10% corresponds to the achromatic sector of a particular hexagonal slice in the HSV model. The range between 10% and 20% represents a sort of transition from the achromatic to the chromatic areas. Selecting $T_{sat} = 20\%$ as a lower bound yields satisfactory segmentation results, however, we have found that the addition of a select number of pixels within the 10–20% range can improve the results. This procedure is outlined below.

A principal polyhedron, PP that corresponds to skin colored clusters with well defined Saturation components is formed by the selection of the following four thresholds

$$T_{hue1} = 340^\circ \leq H \leq T_{hue2} = 360^\circ $$

(4.10)

$$T_{hue3} = 0^\circ \leq H \leq T_{hue4} = 50^\circ $$

(4.11)

$$S \geq T_{sat1} = 20\% $$

(4.12)

$$V \geq T_{vat} = 35\% $$

(4.13)

Although this polyhedron is successful in extracting the skin-tone regions, an improvement can be realized if an additional number of pixels are selected from a second polyhedron, SP. This second polyhedron corresponds to the 10–20% transitional
range and is determined adaptively as described below.

The histogram of all Saturation values that lie within the bounds of equations (4.10, 4.11, and 4.13) is first formed. The analysis of this histogram allows the threshold, $T_{\text{sat}2}$, to be selected which is essentially used to separate the chromatic and achromatic regions within the transitional region. Having determined the Saturation histogram, we search for the first peak, $P_{k1}$, beginning the search from the 0% Saturation level. If the first peak exists at a value greater than 20% then the scene consists of mainly chromatic areas and a choice of $T_{\text{sat}2} = 10\%$ can safely be made. If $P_{k1}$ is within the range 0–20% then the image also contains some achromatic regions (more so if $P_{k1}$ is between 0% and 10%) which must be separated. In this case, the second peak, $P_{k2}$ is detected (i.e. as we move away from $P_{k1}$ in the direction of increasing Saturation) and the in-between valley, $Vl_{1,2}$ is found. If $Vl_{1,2}$ lies in the range 0–20% then the selection $T_{\text{sat}2} = \text{Max}(10\%, Vl_{1,2})$ is made, where Max selects the maximum value of the operand. However, if the valley is greater than 20% then a value of 10% is chosen. A similar procedure has been proposed in [76] for determining an adaptive threshold value in the Saturation component. Thus, the selection of $T_{\text{sat}2}$ is summarized as follows

\begin{align*}
T_{\text{sat}2} &= 10\% & \text{if} & & P_{k1} > 20\% \\
T_{\text{sat}2} &= \text{Max}(10\%, Vl_{1,2}) & \text{if} & & P_{k1} < 20\% \cap 0\% \leq Vl_{1,2} \leq 20\% \\
T_{\text{sat}2} &= 10\% & \text{if} & & P_{k1} < 20\% \cap Vl_{1,2} > 20\% 
\end{align*}

(4.14)

(4.15)

(4.16)

In order to extract the significant peaks and valleys, then the histograms above must be smoothed to removed any meaningless local extrema. For this purpose, we apply the well-known scale space filter [78, 79] where the 1-D Saturation histogram, $f_s(x)$ is convolved with the Gaussian function, $g(x, \tau)$ of zero mean, $(m)$ and standard deviation, $(\tau)$

\begin{equation}
F_s(x, \tau) = f_s * g(x, \tau) = \int_{-\infty}^{\infty} f_s \frac{1}{\sqrt{2\pi} \tau} \exp\left[-\frac{(x - u)^2}{2\tau^2}\right] du
\end{equation}

(4.17)
The peaks and valleys can subsequently be determined by examining the first and second derivatives of $F_s$ above. The valleys are found by

$$\frac{\partial F_s}{\partial x} = 0, \quad \frac{\partial^2 F_s}{\partial x^2} < 0$$  \hspace{1cm} (4.19)

while the peaks can be determined by

$$\frac{\partial F_s}{\partial x} = 0, \quad \frac{\partial^2 F_s}{\partial x^2} > 0$$  \hspace{1cm} (4.18)

The procedure just described is effective in separating the chromatic and achromatic regions.

The second polyhedron, SP can now be formed by using the Saturation threshold, $T_{sat2}$ that was just determined and this is defined by

$$T_{hue_1} = 340^\circ \leq H \leq T_{hue_2} = 360^\circ$$  \hspace{1cm} (4.20)

$$T_{hue_3} = 0^\circ \leq H \leq T_{hue_4} = 50^\circ$$  \hspace{1cm} (4.21)

$$T_{sat2} \leq S \leq 20\%$$  \hspace{1cm} (4.22)

$$V \geq T_{vat} = 35\%$$  \hspace{1cm} (4.23)

The two polyhedra, PP and SP, expressed by equations (4.10 - 4.13), and (4.20 - 4.23), respectively, can now be used to extract the areas that correspond to the skin-tone clusters. The initial color segmentation using the defined polyhedra is summarized in the next section.

### 4.2.5 Segmentation Using the Color Attribute

The overall segmentation technique that we propose is shown in the block diagram of Figure 4.7. It consists essentially of two components: 1) A two-stage color processing
module, and 2) a shape and color analysis module which is implemented in the third and final stage.

The first stage of the procedure is composed of three fundamental blocks as shown in Figure 4.7. In the first of these, all of the pixels in the input image or frame that lie within the Principal Polyhedron, PP are extracted and passed on to the next block. The transformation equations of (4.5 - 4.9) can be used to convert RGB pixel values to the HSV color space. The extent of the Hue range as defined in the Principal Polyhedron was chosen to be quite wide so that a variety of skin types could be modeled. As a result of this, other objects in the scene with "skin-like" colors (i.e. reddish-brown shirt) may also be extracted by this first block. Thus, the function of the second block is to separate these objects by color if this case arises. This is accomplished by analyzing the Hue histogram of the extracted pixels. Scale space filtering, as described earlier, is used to smoothen the histogram and obtain the meaningful peaks and valleys. The valley between two peaks is used to separate two objects that possess different Hue ranges (i.e. the facial region and a different colored object). Each of these objects, O_i, will now be passed through the remaining blocks of the segmentation process. Incidentally, in the remote case that another object matches the skin color of the facial area (i.e. separation is not possible by the scale space filter), then the shape analysis block in Stage 3 will again provide the necessary discriminatory functionality. In the last block of Stage 1, a binary median filter is applied to each object followed by a region filling and removal step. Further details of this final post-processing block are described in the next subsection.

The output from the first stage is next passed on to the second stage of the color processing module. As mentioned earlier, the purpose of this second stage is to refine the segmentation results of the initial stage. In most cases, very reasonable results may be obtained even if this second stage is bypassed. In the first block, the secondary polyhedron, SP, is now used to extract the set of pixels that lie within this solid (the first block of both, Stage 1 and 2 can actually be implemented in one pass of the image). The extracted pixels, S_i are subsequently merged with the results from Stage 1. The merging process is performed as follows. Each pixel, S_i is taken, and the
distance, to the centroid of each object, $d_{cs}$ is computed. If the distance to the closest object is less than a certain threshold, then the pixel under consideration is added to that particular object. The threshold chosen here is that $d_{cs}$ must be within a certain factor, $f_d$ of the distance from the centroid of the object to the most distant point of the object, $d_{cp}$. In other words, $d_{cs} \leq f_d \times d_{cp}$, where a reasonable selection of $f_d$ is made if this factor ranges between 1.0 and 1.5. The merging block just described also consists of binary operations (i.e. performed on the object silhouettes) which can be implemented at a very low computational complexity.

The output from the merging block of the second stage is finally passed on to a post-processing block similar to the one in Stage 1 with the exception of an additional morphological operator. This block essentially refines the shape of the objects in the image and produces the final results from the color processing module. The details of this post-processing block are presented next.

### 4.2.6 Median Filtering and Region Filling/Removal

The median filter has found its way into numerous applications and has been particularly successful in the filtering of noise corrupted images and video sequences [19]. Here, the median filter is applied in the third block of each stage of the color processing module. Once again, we operate on the binary image frames which consist of the object silhouettes. The purpose of this median operation is to smoothen these silhouettes and also eliminate any isolated misclassified pixels that may appear as impulsive type noise from the initial color extraction stage (i.e. the output from block 2 of either stage).

The two-dimensional median filter is given by

$$y_{k,l} = med \{ x_{k+r,l+s} ; \ (r, s) \in A \} \tag{4.24}$$

where $A$ defines the size and structure of the filter window about the central pixel $(k, l)$. A set of $n$ observations, $x_i$, for $i = 1, \ldots, n$ are obtained from the filter
window, and the median value is computed as follows

$$y_{k,l} = \text{med}(x_i) = x_{(\nu+1)}$$

(4.25)

where \(n = 2\nu + 1\) and \(x_{(i)}\) denotes the i-th order statistic. Square filter windows of size 5x5, and 7x7 provide a good balance between adequate noise suppression, and sufficient detail preservation. The binary output, \(y_{k,l}\) above, can also be determined by a simple counting procedure which leads to a fast implementation.

The result of the median operation is successful in removing any misclassified "noise-like" pixels, however, small isolated regions and small holes within object areas may still remain after this step. Thus, we follow the application of median filtering by region filling and removal. This operation fills in small holes within objects which may occur due to color differences (i.e. eyes and mouth of the facial skin region), extreme shadows, or any unusual lighting effects (specular reflections). At the same time, any erroneous small regions are also eliminated as candidate object areas.

This second post-processing step involves boundary extraction and contour tracing/counting of the median filtered binary image. The boundaries or edges of the binary image are easily determined by identifying the black pixels with at least one white nearest neighbor. The edge points of each contour formed are subsequently followed (under eight connectivity) and counted. If the contour boundary is less than a pre-determined threshold then the region is either filled or removed. If the region is surrounded by neighboring skin pixels then it is filled otherwise it is eliminated.

4.2.7 Morphological Processing

The result of median filtering and region filling/removal yields one or more objects of significant size in which one of these is the facial region. In certain video sequences however, we have found gaps or holes around the eyes of the segmented facial area. This occurs in sequences where the forehead is covered by hair and as a result, the eyes fail to be included in the segmentation. Two morphological operators are used in the final block of the color processing module to account for this problem and also
to smoothen the facial contour.

Most morphological operations can be defined in terms of two basic operations, erosion and dilation [80]. The erosion of an object $X$ with the structuring element $B$ is defined as the set of all points $x$ such that $B_x$ (the translation of $B$ so that its origin is located at $x$) is included in $X$,

$$X \ominus B = \{x : B_x \subseteq X\} \quad (4.26)$$

Similarly, the dilation of $X$ by $B$ is the set of all points $x$ such that $B_x$ hits $x$, that is, they have a non-empty intersection [67],

$$X \oplus B = \{x : B_x \cap X \neq \emptyset\} \quad (4.27)$$

The erosion outlined above uniformly reduces the size of an object whereas dilation performs the inverse and expands the object size. When combined, these two operations form the familiar morphological opening,

$$X_B = (X \ominus B) \oplus B \quad (4.28)$$

and closing,

$$X^B = (X \oplus B) \ominus B \quad (4.29)$$

Here, we use these last two operations in the final post-processing stage. The closing operation is first used to fill in small holes and gaps followed by an opening operation which is used to remove small spurs and thin channels. Both of these operations maintain the original shapes and sizes of the object. A compact structuring element such as a circle or square without holes can be used to implement these operations and at the same time it can also help to smoothen the object contours. Furthermore, these binary morphological operations can be implemented by low complexity “hit or miss” transformations [80].

The output from the last block of the second stage, $C_i$, is the final result that
is obtained from the color processing module. At this point, the segmented results may contain one or more objects, $C_i$ in which one of these consists of the facial area. The shape and color analysis module of Stage 3 provides the mechanism to correctly select and classify the facial region. More details of this third stage are provided in the following section on shape and color analysis.

4.3 Shape and Color Analysis

4.3.1 Introduction

The output from the color processing module may contain objects other than the facial area. In this case, additional processing is needed to guarantee that the actual face will be extracted rather than an object with similar hue characteristics. In order to achieve this, a number of expected facial characteristics such as the shape, symmetry, and facial location within the image should be used to determine the correct facial region. These facial characteristics will be fuzzified so that they become less sensitive to variations in the feature values. Although we apply the knowledge-based methodology to the problem of face location, it should be noted that feature-based recognition systems can be used to identify arbitrary objects. Such systems are based on the development of an object description from examples that are available to the designer. In the actual operating phase, the knowledge-based system associates a membership value with every feature for each one of the objects. These values give us an indication of the goodness of fit with an ideal prototype of the corresponding feature. An overall ‘goodness of fit’ value can finally be derived for each object by combining the measures obtained from the individual primitives.

In most cases, the description of an object cannot be characterized by some unique or ideal value. However, fuzzy set theory can be used to quantify the acquired knowledge about the object. A number of fuzzy membership functions can be utilized to transform the physical measurements of the object into a set of values in the interval $[0, 1]$. The value of a particular membership function quantifies the degree to which
the object fits the corresponding primitive. Depending on the construction of the knowledge-based system many of these membership values can be fused together to generate an overall goodness of fit measure for the object under consideration.

In conventional knowledge-based face recognition systems, features such as the width of the eyes, nose and mouth, the distances between pairs of facial components, and the geometry of the human face are used as primitives. In Stage 3 of our segmentation scheme we utilize a set of features that are suitable for our application purposes. In most videoconferencing, or videophone-type sequences, the scene consists of front-view faces which are relatively close to the center of the image. Thus, we utilize features such as the location of the face, its orientation from the vertical axis, and its aspect ratio to assist us with the location/recognition task. Values from these primitives are used to construct the membership functions using a set of examples that are available during the training phase. In the evaluation phase, the corresponding membership function values are used to determine the degree to which each object satisfies the particular invocation of the facial feature.

In the methodology we propose, each segmented object, \( C_i \) (i.e. obtained from the color processing module of Stage 2), is examined to determine the degree to which it satisfies the selected facial primitives. More specifically, we consider the following four features (primitives) in our face localization system:

1. **Deviation from the average Hue value of the different skin-type categories.**
   The average Hue value for different skin-types varies amongst humans and depends on the race, gender, and the age of the person. However, it was shown in the previous section that the facial region exhibits regular properties in the HSV color space. In particular, the Hue values of skin fall within a specific range for all skin-type categories (Table 4.1). The average Hue of the different skin-types forms a range that represents the most probable Hue for human skin-tones. The deviation of an object's expected Hue value from this defined range gives us an indication of its similarity to skin-tone colors.
2. *Face aspect ratio.*

Given the geometry and the shape of the human face, it is reasonable to expect that the ratio of height to width falls within a specific range. If the dimensions of a segmented object fit the commonly accepted dimensions of the human face then it can be classified as a facial area.

3. *Vertical orientation.*

The location of an object in a scene depends largely on the viewing angle of the camera, and the acquisition devices. In video sequences intended for videoconferencing, videophone, or multimedia mail applications, it is assumed that:

(a) The head is not tilted forwards, or backwards so that the face becomes occluded.

(b) Only reasonable rotations of the head are allowed in the image plane. This corresponds to a small deviation of the facial symmetry axis from the vertical direction. This is a logical assumption for the intended applications, as the head will not be parallel to the horizontal axis in a video communication scenario.

This primitive is utilized so that an object is excluded as a valid facial area when its orientation axis (i.e. least moment of inertia) exhibits a large deviation from the vertical axis.

4. *Relative position of the facial region in the image plane.*

By similar reasoning to 3) above, it is more probable that the face will be located in a region that is relatively close to the center rather than the edges of the image. This feature is used so that any segmented objects which are located near the edges, and corners of the image plane are less likely to be classified as facial regions.
4.3.2 Fuzzy Membership Functions

The four features described above are used to define the membership functions required in calculating an appropriate evaluation measure for the invocation of the different primitives. In our segmentation scheme, each membership function provides the degree of similarity of the given object to the facial primitive in question. Thus, the membership values are used to quantify the deviation from the expected, or ideal feature value.

A number of membership function models can be constructed and empirically evaluated. A simplified function model is utilized here in order to keep the complexity of the overall scheme to a minimum. A trapezoidal shape was selected as the membership function for each of the primitives described above. The general form of the function is as follows

\[
\mu(x) = \begin{cases} 
\frac{x-c}{a-c} & \text{, if } c \leq x \leq a \\
1 & \text{, if } a \leq x \leq b \\
\frac{d-x}{d-b} & \text{, if } b \leq x \leq d \\
0 & \text{, otherwise}
\end{cases}
\] (4.30)

This type of membership function attains the maximum value only over a limited range of input values. Symmetric or asymmetrical trapezoidal shapes can be obtained depending on the selected parameter values of \(a, b, c,\) and \(d\). The membership function can assume any value in the interval \([0, 1]\), including both of the extreme values. A value of 0 in the definition above, indicates that the event is impossible. On the contrary, the maximum membership value of 1 represents total certainty. The intermediate values are used to quantify variable degrees of uncertainty. The estimates for the four membership functions are obtained by a collection of physical measurements of each primitive from our extensive database. The values of the trapezoidal parameters in the four membership functions are set so that each function accurately represents the physical primitives observed.

The image database that was constructed for the analysis of the skin-tone distri-
butions was also used in devising the ranges of the trapezoidal membership functions. The Hue characteristics of the facial region (for different skin-type categories) were used to form the first membership function. This function is built using the discrete universe of discourse $[-20^\circ, 50^\circ]$, and the summarized values in Table 4.1. The lower bound of the average Hue observed in the image database is approximately $8^\circ$ (African-American distribution) while the upper bound average value is around $30^\circ$ (Asian distribution). A range is formed using these values, where an object is accepted as a skin-tone color with probability 1 if its average Hue value falls within these bounds. Thus, the membership function associated with the first primitive is defined as follows

$$
\mu_1(x_1) = \begin{cases} 
\frac{(x_1+20)}{28}, & \text{if } -20^\circ \leq x_1 \leq 8^\circ \\
1, & \text{if } 8^\circ \leq x_1 \leq 30^\circ \\
\frac{(50-x_1)}{20}, & \text{if } 30^\circ \leq x_1 \leq 50^\circ 
\end{cases} \tag{4.31}
$$

Experimentation with a wide variety of facial images has led us to the conclusion that the aspect ratio (height/width), $x_2$, of the human face has a nominal value of approximately 1.5. This finding confirms previous results reported in the open literature [57]. However, in certain video sequences we must also compensate for the inclusion of the neck area which has similar skin-tone characteristics to the facial region. This has the effect of slightly increasing the aspect ratio. Using this information along with the observed aspect ratios from our database, we can tune the parameters of the trapezoidal function for this second primitive. The final form of the function is given by

$$
\mu_2(x_2) = \begin{cases} 
\frac{(x_2-0.75)}{0.5}, & \text{if } 0.75 \leq x_2 \leq 1.25 \\
1, & \text{if } 1.25 \leq x_2 \leq 1.75 \\
\frac{(2.25-x_2)}{0.5}, & \text{if } 1.75 \leq x_2 \leq 2.25 \\
0, & \text{otherwise}
\end{cases} \tag{4.32}
$$

The vertical orientation of the face in the image is the third primitive used in our shape recognition system. As mentioned previously, the orientation of the facial area (i.e. deviation of the facial symmetry axis from the vertical axis) is more likely to be
aligned towards the vertical due to the type of applications considered. The following range of values were observed for the orientation of the facial region in over 100 frames from several video sequences: i) \((0^\circ - 10.5^\circ)\) in Foreman, ii) \((0^\circ - 4.75^\circ)\) in Akiyo, iii) \((3.5^\circ - 21^\circ)\) in Carphone, and iv) \((0.5^\circ - 6.75^\circ)\) in Claire. A reasonable threshold selection of 30° can be made for valid head rotations as confirmed in the observed sequences above. Thus, a membership value of 1 is returned if the orientation angle is less than this threshold. The orientation, \(x_3\), is found by

\[
x_3 = \frac{1}{2} \tan^{-1} \left[ \frac{2\mu_{1,1}}{\mu_{2,0} - \mu_{0,2}} \right]
\]

(4.33)

where \(\mu_{p,q}\) are the \((p, q)\) order central moments

\[
\mu_{p,q} = \sum_{(k,l) \in \mathcal{R}} (k - \bar{k})^p (l - \bar{l})^q
\]

(4.34)

and \((\bar{k}, \bar{l})\), is the center of mass

\[
\bar{k} = \frac{1}{N} \sum_{(k,l) \in \mathcal{R}} k
\]

(4.35)

\[
\bar{l} = \frac{1}{N} \sum_{(k,l) \in \mathcal{R}} l
\]

(4.36)

In the above equations, the object under consideration is represented by the region, \(\mathcal{R}\) containing \(N\) pixels. As a result of this, the membership function for this primitive can be defined as follows

\[
\mu_3(x_3) = \begin{cases} 
1, & \text{if } 0^\circ \leq x_3 \leq 30^\circ \\
\frac{(90 - x_3)}{60}, & \text{if } 30^\circ \leq x_3 \leq 90^\circ 
\end{cases}
\]

(4.37)

The last primitive used in our knowledge-based system refers to the relative position of the face in the image. Due to the nature of the applications considered, we would like to assign a smaller weighting to objects that appear closer to the edges and corners of the images. For this purpose, we construct two membership functions.
The first one returns a confidence value for the location of the segmented object with respect to the X-axis. Similarly, the second one quantifies our knowledge about the location of the object with respect to the Y-axis. The discrete universe of discourse for these membership functions depends on the dimensions of the image. Since our system supports variable size images, the following membership function has been defined for the position of the segmented object with respect to either the X or Y-axis

\[ \mu_4(x_4) = \begin{cases} \frac{(x_4 - d)}{d}, & \text{if } d \leq x_4 \leq \frac{3d}{2} \\ 1, & \text{if } \frac{3d}{2} \leq x_4 \leq \frac{5d}{2} \\ \frac{(3d - x_4)}{d}, & \text{if } \frac{5d}{2} \leq x_4 \leq 3d \\ 0, & \text{otherwise} \end{cases} \]  

The membership function for the X-axis is determined by letting \( d = \frac{D_x}{4} \) where \( D_x \) represents the horizontal dimensions of the image (i.e. in the X-direction). In a similar way, the Y-axis membership function is found by letting \( d = \frac{D_y}{4} \) where \( D_y \) represents the vertical dimensions of the image (i.e. in the Y-direction). Thus, the X, and Y-axis membership functions are assigned the maximum value if the centroid of the object is within a window that is relatively central to the image. The parameter values have been appropriately chosen for the intended applications.

4.3.3 Aggregation Operators

In the end, the individual membership functions must be combined to form an overall decision. A nonlinear operator is used to arrive at this final decision by appropriately combining the information from the different features. The function of the operator is to reduce the imprecision and uncertainty in the decision making process. A number of fuzzy operators can be used to combine or fuse together the various sources of information. Conjunctive type of operators represent a consensus between the different sources of information. Such operators search for a simultaneous satisfaction of the various primitives or objectives by weighting more heavily the criterion with the smallest membership value. On the contrary, disjunctive operators express re-
dundancy between information by assigning the most weight to the criterion with the largest membership value. Compromise operators, such as weighted mean operators or fuzzy integrals provide a trade-off among different, and possibly incompatible objectives.

The latter approach is followed in this paper. An aggregator, (fuzzy connective) whose shape is defined apriori, is used to combine the four elemental membership functions resulting from the primitives discussed above.

The *compensative operator* selected mixes both conjunctive and disjunctive behavior. Following the results in [81], the operator is defined as the weighted mean of a *(logical AND)* and a *(logical OR)* operator

\[ A \bigcirc_\gamma B = (A \cap B)^{1-\gamma} \cdot (A \cup B)^\gamma \]  
(4.39)

where \( A \), and \( B \) are sets defined on the same space and represented by their membership functions. Different *t-norms* and *t-conorms* can be used to express a conjunctive or a disjunctive attitude. If the product of membership functions is utilized to determine the intersection *(logical AND)* and the possibilistic sum for the union *(logical OR)*, then the form of the operator becomes as follows [81]

\[ \mu_c = \prod_{j=1}^{m} \mu_j^{(1-\gamma)} \cdot (1 - \prod_{j=1}^{m} (1 - \mu_j))^\gamma \]  
(4.40)

where \( \mu_c \) is the overall membership function which combines all the knowledge primitives for a particular object, and \( \mu_j \) is the \( j^{th} \) elemental membership value associated with the \( j^{th} \) primitive. The weighting parameter \( \gamma \) is interpreted as the *grade of compensation* taking values in the range of \([0, 1]\) [81]. The product and the possibilistic sum are not the only operators that can be used in (4.39). A simple and useful *t-norm* function is the *min* operator while the corresponding one for the *t-conorm* is the *max* operator. In this paper, we utilize this *t-norm* to represent intersection. In
this case, the compensative operator of (4.39) has the following form

\[ \mu_c = \left( \min_{j=1}^{m} \mu_j \right)^{(1-\gamma)} \left( \max_{j=1}^{m} \mu_j \right)^{\gamma} \] (4.41)

The form of the compensative operator in (4.40) is not unique. A number of other mathematical models can be used to represent the AND aggregation. An alternative operator, which combines the averaging properties of the arithmetic mean (member of the averaging operator class) with a logical AND operator (conjunctive operator) was also proposed in [81]

\[ \mu_c = \gamma \min_{j=1}^{m} \mu_j + (1 - \gamma) (m^{-1} \sum_{j=1}^{m} \mu_j) \] (4.42)

where \( \mu_c \) is again the overall membership function and the parameter \( \gamma \in [0, 1] \) is interpreted as the grade of compensation. In this equation the \( \min \) \( t \)-norm stands for the logical AND. Alternatively, the product of membership functions can be used instead of the \( \min \) operator in the above equation. The arithmetic mean is used to prevent higher elemental weights with extreme values to dominate the final outcome.

Compensatory operators are intuitively attractive and provide a simple yet powerful method to express the interactions between different knowledge primitives. For this reason, our shape and color analysis module utilizes these operators in correctly selecting the facial area from a set of candidate objects.

In this work we define \( \gamma = 0.5 \). Therefore, the compensative operator assumes the form of a weighted product. The \( \min \) and \( \max \) operators were selected to model the corresponding \( t \)-norm and \( t \)-conorm functions [82]. Thus, the overall fuzzy membership function can be defined as

\[ \mu_c = \left( \left( \min_{j=1}^{m} \mu_j \right) \left( \max_{j=1}^{m} \mu_j \right) \right)^{0.5} \] (4.43)

In general, additional weighting factors must be used in the generalized function above in order to absorb possible scale differences in the definition of the elemental membership functions. However, all the elemental membership functions used here
are within the interval \([0, 1]\), and thus, no such weighting factors are required.

The aggregation operator defined in (4.43) can be used to form the final decision based on the four primitives under consideration. However, in order for our results to be meaningful, the nonlinear operator applied must satisfy some properties that will guarantee that its application will not alter in any manner the elemental decisions about the knowledge primitives. In the literature, there are a number of properties that all the aggregation or compensative operators must satisfy. We will try to examine if the operator which we intend to use in the calculation of the final membership function satisfies these properties \([83]\). These properties are listed below:

1. **Convexity:**
   The convexity of the operators allows for a compromise among the different elemental membership functions. The weighted operator in (4.43) is convex since it is known from statistics that
   \[
   \mu_c = (\min_{k=1}^j \mu_k \max_{k=1}^j \mu_k)^{0.5},
   \]
   \[
   \min_k \mu_k \leq \mu_c \leq \max_k \mu_k,
   \]
   where \(k = 1, 2, ..., j\) is the number of elemental membership functions to be fused together.

2. **Neutrality (Symmetry):**
   The operator used here is symmetric. The property guarantees that the order of presentation for the elemental membership functions does not affect the overall final membership value. It is not hard to see that by simply interchanging the order of presentation for the \(\text{max}\) and the \(\text{min}\) value the same result will occur
   \[
   \mu_c = (\max_{j=1}^m \mu_j)(\min_{j=1}^m \mu_j)^{0.5} = (\min_{j=1}^m \mu_j)(\max_{j=1}^m \mu_j)^{0.5}
   \]
   \[
   (4.46)
   \]

3. **Monotonicity:**
   The property of monotonicity guarantees that the stronger piece of evidence
(larger elemental membership value) generates a stronger support in the final membership function.

Let us assume that $\mu_i \leq \mu_i$, with $A = \min_{k=1}^{j} \mu_k$ and $B = \max_{k=1}^{j} \mu_k$.

By the definition of the $\text{min}$ and $\text{max}$ operators

$$\min(A, \mu_i) \leq \min(A, \mu_i)$$  \hspace{1cm} (4.47)

and

$$\max(A, \mu_i) \leq \max(A, \mu_i)$$  \hspace{1cm} (4.48)

Therefore,

$$\left(\min(A, \mu_i)\max(A, \mu_i)\right)^{0.5} \leq \left(\min(A, \mu_i)\max(A, \mu_i)\right)^{0.5}$$  \hspace{1cm} (4.49)

4. **Idempotence:**

The operator considered in (4.43) is idempotent. The property guarantees that the outcome of the overall function generates the same value with each elemental value if all of them report the same result. Given the form of the operator

$$\mu_c = (\mu_a \mu_b)^{0.5} = (\mu^* \mu^*)^{0.5} = \mu^*$$  \hspace{1cm} (4.50)

with

$$\mu_a = (\min_{j=1}^{m} \mu_j) = \mu^*$$  \hspace{1cm} (4.51)

if $\mu_1 = \mu_2 = \ldots = \mu_j = \mu^*$.

In summary, it is proven that the compensatory operator that we intend to utilize for our shape and color analysis module in Stage 3 corresponds to an aggregation class which satisfies a number of natural properties, such as neutrality and monotonicity.
4.3.4 Metadata and Content-Based Retrieval

Thus far in this chapter, we have outlined a procedure for automatically extracting the facial region in videophone-type sequences. This allows for more effective compression schemes that do not suffer from the unpleasant artifacts associated with block-based techniques. However, the segmentation of a scene into objects also permits sophisticated storage and retrieval schemes based on the contents of the information.

Multimedia data such as audio, still images, and video are binary in nature, and hence, un-interpreted. The appropriate interpretations of different media objects (i.e. audio, images, video) must be constructed in order to permit storage and retrieval based on content. The generation of these interpretations, termed metadata, is achieved by applying a set of feature extracting functions on the various media objects [84, 85]. These functions are media dependent and may also be content dependent and/or independent for each media type. As the name implies, content-dependent metadata depends only on the content of the media objects. For a static image media type, this may be the derivation of the facial features of a person within the image (i.e. color of hair and skin, type of nose or mouth, etc.). Metadata that is content-independent does not rely on the media information but is closely associated with it. This can be the copyright owner, or the creation time of the image if we refer to the previous example. Here, we briefly examine the metadata generation process specifically for facial images which can be incorporated within the framework of future MPEG-7 video applications.

The algorithms used for generating the required metadata are unique to the type of images being analyzed (i.e. facial images, satellite images, etc.). The following four steps are generally involved in extracting the features from image object types: 1) Object locator design, 2) Feature selection, 3) Classifier design, and 4) Classifier training [84]. It is the function of the Object locator to segment the image into regions or objects that possess distinct, identifiable features. In the case of facial images, the segmentation of the facial region is of primary significance. The proposed technique outlined in this chapter can be used to achieve the objectives set out in this first step.
The algorithm in Figure 4.7 automatically, and robustly locates and tracks the facial area within an image or video sequence.

Once the image has been appropriately segmented, then the different objects must be classified according to the desired features. The purpose of the Feature selection step is to determine the important and necessary features of the various objects. For facial images, these features may include skin or hair color, fuzzy descriptions of various features such as the nose or mouth, their location within the image, and their spatial relationships with other objects in the scene. An $n$-dimensional feature vector $\mathbf{f} = (f_1, \ldots, f_n)$ can subsequently be devised, where each metadata feature, $f_i$ represents a certain primitive of a particular facial image. The vector, $\mathbf{f}$ can be utilized to describe the overall facial image or a set of similar featured ones. The representation of metadata in this way allows one to use the metadata feature space as a search space for extracting images when a user query is comprised of image features. It is the function of the Classifier design and training stages to select the appropriate features, store the metadata in suitable index structures, and establish a mathematical basis for retrieving data based on user queries. The latter three steps of the metadata generation process are beyond the scope of this thesis, but it gives us an indication of the enormous potential for content-based access to the vast sources of digital information and the tremendous work that lies ahead.

### 4.4 Experimental Results

The steps outlined in Figure 4.7 were used to locate and track the facial region of several videophone-type sequences. The results from 3 CIF and 2 QCIF sized sequences, as well as three still images, are presented below: 1) Claire, 2) Miss America, 3) Akiyo, 4) Foreman, 5) Carphone, 6) an African-American sample image, and 7) two images from the M2VTS (Multi Modal Verification for Teleservices and Security Applications) Multimodal Face Database of the European Acts Project 102 (http://ns1.tele.ucl.ac.be/M2VTS/), respectively. The aforementioned test sequences and images were chosen so as to represent all skin-type categories (i.e. Caucasian,
African-American, and Asian), various types of motion within the scene (camera pan and zoom, head rotations and tilts, and moving complex backgrounds), and facial images with glasses and hats (i.e. from the M2VTS database). The segmentation results in Figure 4.15 illustrate the robustness of the technique to the various cases of motion, skin color, and obstructive objects (i.e. glasses, hats, etc.) mentioned above. The facial region is successfully located and tracked when the head is rotated as in Figure 4.15 a) or in the cases of head tilts as in Figures 4.15 c) and f). The technique is successful even when the facial area undergoes various deformations caused by different facial expressions (i.e. Claire, Miss America, Carphone, and Akiyo). The Foreman sequence of Figure 4.15 d) demonstrates that the extraction process is invariant to pans and zooms within the scene while the Carphone sequence in 4.15 f) illustrates the effectiveness of the algorithm under conditions of a complex and moving background. The latter scenario may be the case in an environment where mobile videophones are employed. The robustness of the algorithm is also exemplified in Figures 4.15 g) and h) where obstructive objects such as glasses or hats do not affect the extraction of the facial region. Finally, we observe that successful results are obtained for the complete range of skin colors (Akiyo, African-American sample, and Miss America, i.e. Figures 4.15 c), e), and b), respectively).

In Figures 4.8 a) through f) we present the Hue histograms of Frame 20 from the different video sequences as well as the African-American sample image.

These are obtained by passing each of the images through the Principal Polyhedron (PP) as defined earlier. The smoothed scale-space filtered versions of these histograms are also shown alongside the former, and are derived from the second block in Stage 1 of the color processing module (Figure 4.7). A standard deviation of $\sigma = 2$ in the Gaussian function, $g(x, \sigma)$ provided adequate smoothing of the histograms and was found to be appropriate for the different skin-tone distributions which had standard deviations, $\sigma$ that ranged from $5.1^\circ - 8.2^\circ$. In the Claire sequence of Figure 4.8 a), the histograms indicate that one distinct Hue range exists which has a mean value around $34^\circ$. In turn, this range contains only one distinct object, $O_1$ which is the facial region. The observed Hue values are shifted towards
the yellow spectrum which is also evident visually from the results in Figure 4.15 a). Thus, we see that in the case of the Claire sequence, the shape and color analysis module (SC module) of Stage 3 need not be invoked.

The histograms in both, Figures 4.8 b), and c) indicate that two different Hue ranges exist and we refer to each of these as Hue Regions, HR1 and HR2. Incidentally, each Hue Region, HRi, may contain one or more disjoint areas which we refer to as objects, Oi (Ci is the post-processed version of Oi). The latter, are processed by the SC module in the selection of the facial region. Each Hue range in both, 4.8 b) and c), is determined by utilizing the Hue histogram. Each range, in turn, may contain one or more objects which are analyzed by the shape module in order to correctly select the one which corresponds to the facial area. A more detailed analysis of the three sequences (Akiyo, Miss America, and Foreman) which require processing by the shape and color analysis module is presented in the subsections below.

In Figure 4.8 f), the African-American sample image is shown which consists of only one Hue Region, HR1, containing one object, C1, with a mean of approximately 10°. This value is close to the mean of 8.6° which was found earlier for the African-American distribution. Once again, the shape module is not necessary in the segmentation of this image.

Finally, in the Carphone sequence of Figure 4.8 f) we can only identify one region with a distinct hue range, HR1, (i.e. only one distinct maximum or minimum point) and this has a mean value of approximately 24°. This value is also in accordance with the expected value of the Caucasian distribution and the pixels about the peak belong to the facial area. Some of the values in the tail of the distribution (i.e. 35 – 50°) do not correspond to the facial area, however, these pixels are scattered and thus are removed by median filtering and region removal. As a result, only one object, C1 remains which is the extracted facial region. Incidentally, we have found that this sequence benefits by using the Secondary Polyhedron, (SP in Stage 2 of Figure 4.7). In Figure 4.9, the importance of the Secondary Polyhedron and the appropriate Saturation threshold of the Principal Polyhedron is illustrated. In Figure 4.9 a), Frame 20 of the original Carphone sequence is shown while in part b), the extracted facial region using the
PP and SP stages as proposed in Figure 4.7 is displayed. In part c), the results are shown when only the PP Stage of the color processing module is utilized (i.e. pixels with Saturation values between 10 and 20% are not considered). In this case, the segmentation is not as effective as the two-stage process since the left side of the facial region has not been extracted completely. A significant portion of the facial area contains pixels with Saturation values in the transitional chromaticity region mentioned earlier. This is a result of excessive light within the scene (i.e. exposure of sunlight on that side of the facial region) causing the skin tones to have lower Saturation values and appear slightly washed out. In part d), however, we have an oversegmented image when the Saturation threshold, \( T_{sat} \), of the PP is lowered to 15% from the 20% value set in equation 4.12. Thus, the effectiveness of the proposed two-stage extraction process outlined in Figure 4.7, and the establishment of the appropriate Saturation thresholds is clearly evident in this example. A more detailed account of the Akiyo, Miss America, and the Foreman sequences is given next.

### 4.4.1 Akiyo Sequence

The set of results in Figure 4.10 illustrate the details of the segmentation process for the Akiyo sequence (Frame 20) through the different stages of the facial extraction scheme outlined in Figure 4.7.

The segmentation process begins by first passing the input image through the Principal Polyhedron, PP. The histogram of the Hue values within PP is formed and subsequently smoothened by the scale-space filter, \( g(x, \tau) \). The results of the histograms obtained from this step are shown in Figure 4.8 c). From the scale-space filtered version we can identify two Hue Regions. The minimum value at \( H = 18^\circ \) is used to separate the two regions as follows \( 18^\circ \leq HR_1 \leq 50^\circ \) and \( -20^\circ \leq HR_2 < 18^\circ \). The local maxima and minima in Figure 4.8 c) are determined automatically by the scale-space filtering technique described earlier. The images in Figures 4.10 a) to f) illustrate the results obtained from the remaining steps in the segmentation process for the region \( HR_1 \) which, incidentally corresponds to the facial area. A similar procedure is also carried out for the second region, \( HR_2 \).
In Figure 4.10 a) the results are shown for HR₁ after the primary extraction process using PP. Most of the facial skin area is extracted from this initial step, with the addition of some erroneous regions from the jacket area. Figure 4.10 b) illustrates the output after median filtering which is used to remove the isolated "noise-like" pixels. A filter window of 7×7 was chosen for the CIF size images while a 5×5 mask was utilized for the smaller QCIF size format. In 4.10 c) the results are shown after region filling/removal. This step eliminates small misclassified regions and also fills in holes within the larger regions (i.e. the eyes, and mouth). A region was removed if its perimeter was less than a threshold value of 200 pixels for the CIF images and 100 for the QCIF sequences. These values were found to be appropriate choices for the videophone-type applications that were considered. A selection of smaller thresholds can also be made for other applications (i.e. where the face occupies a very small area within the image), however, this requires the SC module to analyze a greater number of objects. At this point in the segmentation process, we are left with one object, O₁ within the first Hue Region, HR₁.

The steps highlighted above mark the completion of Stage 1 of the color processing module. In the first block of Stage 2, the pixel values within the Secondary Polyhedron, SP are extracted. As mentioned earlier, this is done to include pixels which lie in the transition range of chromatic and achromatic regions. The threshold, \( T_{sat2} \) found in equations (4.20 – 4.23) defines SP, and its selection is made from the Saturation histogram formed by equations (4.10, 4.11, and 4.13). This histogram is shown in Figure 4.11 a) for the region of HR₁, while the one in 4.11 b) illustrates that for HR₂. A choice of \( T_{sat2} = 18\% \) is made for SP of HR₁ according to the conditions in equation (4.15). Figure 4.10 d) shows the pixels extracted by the Secondary Polyhedron while 4.10 e) displays the result of the merging block in the second step of Stage 2. A factor of \( f_d = 1.1 \) was used in merging the results from the two polyhedra. As we can see, in the Akiyo sequence, the SP extraction process has virtually no effect in refining the results obtained from the the Principal Polyhedron. Finally, in Figure 4.10 f) the segmentation results are shown for the region HR₁ after the final post-processing block in Stage 2. A semi-circular structuring element (SCSE) of ra-
radius 15 was utilized for the morphological closing operation and a radius of 5 for the opening operation. This was found to be quite effective in accomplishing the desired objectives. Furthermore, the SCSE performed equally as well as its circular counterpart while requiring only half the number of operations. In Figure 4.10 f), only one object remains, $C_1$ (i.e. $C_i$ is the post-processed version of $O_i$) and this happens to be the facial region. A similar step-by-step procedure was repeated for the second Hue Region, $HR_2$, and in this case the object $C_2$ in Figure 4.10 h) was obtained.

The two objects, $C_1$ and $C_2$ obtained from the color module in Figure 4.7 were subsequently passed on to the shape and color analysis module for the selection process. The shape module analyzes each object and computes a set of values for the different primitives considered. Table 4.2 summarizes the results of the five primitives along with the membership function values, $\mu_i$, for $i = 1, \ldots, 5$ for each of these features.

**Table 4.2: Akiyo (Width×Height=352×288) Shape & Color Analysis.**

<table>
<thead>
<tr>
<th>Object</th>
<th>Centroid Location</th>
<th>Orientation</th>
<th>Object Ratio</th>
<th>Mean Hue</th>
<th>Aggregation</th>
</tr>
</thead>
<tbody>
<tr>
<td>$C_i$</td>
<td>X</td>
<td>$\mu_1$</td>
<td>$\mu_2$</td>
<td>$\theta^\circ$</td>
<td>$\mu_3$</td>
</tr>
<tr>
<td>1</td>
<td>178</td>
<td>1</td>
<td>134</td>
<td>1</td>
<td>3.10</td>
</tr>
<tr>
<td>2</td>
<td>246</td>
<td>0</td>
<td>245</td>
<td>0.2</td>
<td>31.70</td>
</tr>
</tbody>
</table>

The aggregation of these functions, $\mu_c$, is computed by equation (4.43) and is shown in the final column of the table. The first object, $C_1$ scored the highest aggregate value and therefore, was selected as the facial region. A high membership value was obtained for every primitive, except for the Object Ratio which begun to exceed the bounds of the allowable facial aspect ratio. Object, $C_2$ scored reasonably well in Orientation, Object Ratio, and Mean Hue, however, its poor location in the image brought its aggregation value down. Both of these objects, $C_1$ and $C_2$ can be seen in Figures 4.10 g) and h), respectively.
4.4.2 Miss America Sequence

The detailed procedure just described was also applied to the Miss America sequence. The scale-space filtered Hue histogram in Figure 4.8 b) indicates that two Hue Regions exist just as in the Akiyo sequence. These are easily separated into the following two ranges, $8^\circ \leq HR_1 \leq 50^\circ$ and $-20^\circ \leq HR_2 < 8^\circ$. In the first region, $HR_1$, only one object remains (i.e. the facial region) and this is shown in Figure 4.12 b). However, the second region, $HR_2$, contains two objects, $C_2$ and $C_3$, and these are illustrated in Figure 4.12 c) and d), respectively. These latter two correspond to the jacket area and are in the Red-Magenta sector of the Hue hexagon. Figure 4.12 a) (image before the morphological operation) is shown simply to illustrate the importance of the morphological operation in filling the holes around the eye regions in cases where the hair is close to these areas.

The shape and color feature values are provided in Table 4.3 for each of the three objects in the Miss America sequence.

Table 4.3: Miss America (Width×Height=360×288): Shape & Color Analysis.

<table>
<thead>
<tr>
<th>Object</th>
<th>Centroid Location</th>
<th>Orientation</th>
<th>Object Ratio</th>
<th>Mean Hue</th>
<th>Aggregation</th>
</tr>
</thead>
<tbody>
<tr>
<td>$C_1$</td>
<td>X</td>
<td>$\mu_1$</td>
<td>$\mu_2$</td>
<td>$\theta^\circ$</td>
<td>$\mu_3$</td>
</tr>
<tr>
<td>1</td>
<td>177</td>
<td>1</td>
<td>1</td>
<td>4.92</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>245</td>
<td>0</td>
<td>1</td>
<td>47.74</td>
<td>0.7</td>
</tr>
<tr>
<td>3</td>
<td>244</td>
<td>0</td>
<td>0.02</td>
<td>44</td>
<td>0.77</td>
</tr>
</tbody>
</table>

Once again, object #1, $C_1$ is correctly chosen as the facial region based on the computed aggregation value. The objects, $C_2$ and $C_3$, scored poorly in their location and Mean Hue value, and also had lower membership values in the Orientation primitive. The net effect of this was to bring the aggregation value of the erroneously detected objects down to zero in each case.
4.4.3 Foreman Sequence

The histogram of the Foreman sequence in Figure 4.8 d), contains three different Hue Regions (i.e. 3 distinct peaks), which are easily separated by the valleys at 32° and 38°. The first two regions $-20° \leq HR_1 \leq 32°$, $32° \leq HR_2 \leq 38°$, each contain one object, $C_1$ and $C_2$, respectively, while the third region $38° \leq HR_3 \leq 50°$ contains two objects, $C_3$ and $C_4$. In the final analysis, the shape module correctly selects the first object, $C_1$ as the facial area of the sequence. The mean value of $C_1$ (i.e. first peak in Figure 4.8 d) is approximately 25° which is also in accordance with the mean Hue value of the Caucasian skin-type distribution ($\mu = 25.3°$). A summary of the results from the SC module are presented in Table 4.4.

Table 4.4: Foreman (Width $\times$ Height = 176 $\times$ 144): Shape & Color Analysis.

<table>
<thead>
<tr>
<th>Object</th>
<th>Centroid Location</th>
<th>Orientation</th>
<th>Object Ratio</th>
<th>Mean Hue</th>
</tr>
</thead>
<tbody>
<tr>
<td>$C_i$</td>
<td>$X$ $\mu_1$ $Y$ $\mu_2$ $\theta^o$ $\mu_3$ $r$ $\mu_4$</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>76 1 82 1 12.8 1 1.46 1 25 1</td>
<td>1.0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>130 0 159 0 81.6 0.14 1.20 0.9 32 0.9</td>
<td>0.0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>59 0 36 0 82.0 0.13 1.75 1 45 0.25</td>
<td>0.0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>94 0 137 0 64.43 0.43 2.14 0.22 45 0.25</td>
<td>0.0</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

As mentioned previously, four objects are identified in the sequence as a result of three Hue Regions. The images of the four different objects are shown in Figure 4.13. The first object, $C_1$ in part b) corresponds to the facial region and this one is also selected by our knowledge-based system due to its high aggregation value. The remaining objects shown in parts c) and d) of the figure are rejected by the shape and color analysis module due to their poor matching with the constructed facial primitives.

As a final note, we would like to mention once again that the proposed segmentation technique is not affected by the case of glasses on a subject. The pixels which are not extracted by the glasses are filled in by the post-processing operations. A pair of thin-framed glasses can easily be handled by the median filter while thicker-framed glasses are taken into account by the morphological operations. In Figure 4.14 b) we
see that the glasses are not included in the initial extraction, however, they pose no problem in the final output as shown in part c).

4.5 Conclusions

In this chapter, a novel technique was proposed for the automatic location and tracking of the facial area in color video sequences. The attributes of color and shape were utilized in devising a three-stage segmentation scheme which consisted of a two-stage color processing unit, and a single-stage shape/color analysis module. The suggested method led to a consistent and accurate localization of the facial region and performed robustly for different skin-types and various cases of object or background motion within the scene. The first stage of the color processing module was used to extract the regions in the image that matched the Hue characteristics of skin-tones. This extraction process was formulated in the perceptual HSV color space by utilizing the apriori knowledge of the skin-tone distributions for various skin-type categories. The second stage in the color module was essentially used to refine the results of the initial extraction stage. In most cases, it was found that reasonable output could be obtained by excluding this second stage, thereby, decreasing the overall execution time of the algorithm. A number of binary post-processing operations were also included in the color processing unit to refine the shape of the segmented facial region. The computational complexity of these steps were minimal due to the binary nature of the operations. In many cases, only the facial area was extracted from the image, since no other objects in the scene possessed Hue characteristics that were similar to the face. In a situation where more than one object was detected, then the final shape and color analysis stage provided the mechanism to correctly select the facial area. A compensative aggregation operator was used to combine the results from a series of fuzzy membership functions that were tuned for videophone-type applications. A number of features such as object shape, orientation, location, and average Hue were used to form the appropriate membership functions. The three-stage segmentation process appears to be quite promising and can be used with an additional feature
extraction stage to provide higher level descriptions in future video coding environments and content-based retrieval schemes. The facial extraction technique developed in this chapter is next incorporated in an object-based, temporal prediction scheme.
Figure 4.5: Skin color distributions of different races in the HSV space, a) Caucasian, b) African-American, and c) Asian test samples.
4.6 (a)
Figure 4.6: HSV hexcone model at different values of the V component, a) $V=100\%$, b) $V=63\%$, c) $V=T_{val}=35\%$, and d) $V=20\%$. 
Figure 4.7: Overall Segmentation scheme using Color and Shape attributes.
(a)

(b)

4.8 (c)
Figure 4.8: Hue values of the Principal Polyhedron along with its Scale-Space filtered version for Frame 20 of the following video sequences: a) Claire, b) Miss America, c) Akiyo, d) Foreman, e) African-American sample image, and f) Carphone.
Figure 4.9: **Effect of the SP and the Saturation threshold of the PP on the segmentation process:** a) Original Frame 20 of the Carphone sequence, b) results using the PP and SP Stages in the segmentation process as proposed in Figure 4.7, c) output obtained when using only the PP Stage of the extraction process, and d) the results found when the Saturation threshold in the PP is reduced to a value of 15%.
Figure 4.10: Extraction of the facial region for the Akiyo sequence (Frame 20) through the various stages, a) Initial extraction by the Principal Polyhedron for the Hue Region, $8^\circ \leq HR_1 \leq 50^\circ$ b) Median filtered result, c) Region filling and removal, d) Extraction of the Secondary Polyhedron for HR$_1$, e) Region merging of the extracted PP and SP regions, f) Final result of HR$_1$ after morphological processing (i.e. C$_1$), g) Shape processing of object, C$_1$ (i.e. facial region), and h) Shape processing of object, C$_2$ found from the second Hue Region $-20^\circ \leq HR_1 \leq 8^\circ$. 
Figure 4.11: Saturation components of the Principal Polyhedron for the two Hue Regions, HR_1 and HR_2, of the Akiyo sequence (Frame 20).
Figure 4.12: Extraction of the facial region for the Miss America sequence (Frame 20), a) Extraction of the facial region prior to morphological processing, b) Shape processing of object, $C_1$ (i.e. facial region) in Hue Region, $HR_1$, c) Shape processing of object, $C_2$ in Hue Region, $HR_2$, and d) Shape processing of object, $C_3$ in $HR_2$. 
Figure 4.13: Extracted objects to be analyzed by the shape and color analysis module: a) Original Frame 20 of the Foreman sequence, b) the object in Hue Region, HR₁ (i.e. facial region), c) the object in Hue Region, HR₂, and d) the two candidate objects in Hue Region, HR₃.

Figure 4.14: The effects of glasses on the facial extraction process: a) Original Image #1 from the M2VTS database, b) initial extraction by the Principal Polyhedron, and c) final segmented image.
Frame 20

Frames 20-120

4.15 (b)
Figure 4.15: Location and tracking of the facial region for the following images and video sequences: a) Claire, b) Miss America, c) Akiyo, d) Foreman, e) African-American sample image, f) Carphone, g) M2VTS Image #1, and h) M2VTS Image #2.
Chapter 5

Motion Compensated Prediction

5.1 Introduction

Video compression is a key enabling technology for newly emerging digital video applications. Typical image sequences exhibit a significant degree of spatial, and temporal redundancies due to the strong correlation between neighboring pixels, and the similarity amongst adjacent frames, respectively. Exploiting these redundancies can dramatically reduce the amount of information required for storage or transmission purposes. The selection of an appropriate compression scheme depends largely on the coding requirements such as the quality of service, the coding delay, and more recently, the provision for content-based interactivity. Thus, a coder/decoder (codec) must effectively combine these spatial and temporal redundancies to form an efficient compression system, particularly in low bit-rate environments.

Coding techniques that are devised for still images are referred to as intraframe compression schemes and these effectively utilize the spatial redundancies mentioned earlier. Conventional schemes operate in a block-wise manner, that is where the image is made up of smaller, non-overlapping, square blocks. The internationally accepted ITU - CCITT (vital in the world of fax machines), and JPEG image compression standards belong to these intraframe-based methods [86]. Psychovisual redundancies are also present within static images due to the fact that the human eye is insensitive to certain spatial frequencies.
An elementary approach that can be used to compress a video sequence is to apply the intraframe techniques on a frame by frame basis. However, the compression efficiency can be significantly improved if the strong correlation between neighboring frames is taken into account. Interframe compression methods are used to exploit these temporal redundancies that are present within image sequences.

In typical video sequences the change between successive frames is often the result of motion in the scene. This motion may be global due to camera motion (pan and/or zoom), or local due to object motion. The temporal correlation between frames is highest along this trajectory of motion. Motion compensated prediction (MCP) is a technique that is extensively used to exploit this latter property. The intensity values in a particular frame are predicted using the values of another frame, termed the reference frame, along with the appropriate motion trajectories between the two frames. Conventional motion compensated prediction methods rely on standard block matching approaches where displacement vectors are estimated over rectangular blocks of the image. This approach is favorable due to its straightforward approach, however, it fails to adequately model object motion which is non-translatory (i.e. object rotation, deformation, or change of scale). This scheme also suffers from annoying blocking artifacts when components of an image feature are assigned different motion vectors. In order to alleviate these problems, several approaches based on digital image warping have recently been introduced [87, 88, 89, 90, 91]. In these methods, the predicted frames (also referred to as the current frames) are formed by geometrically transforming or warping the previous frames. These techniques have shown significant quality improvements over the conventional methods.

International coding standards such as H.261, and MPEG 1 and 2, belong to the block-based intra and interframe methods described above. These compression schemes take an information-theoretic-based approach, that is, by exploiting the stochastic properties of the inherent signals. Research in this area has matured and reached the limits of compression within this framework. Furthermore, these schemes deal with video exclusively at the frame level, thereby preventing the manipulation of individual objects within the bitstream. Recently, however, greater attention has
been paid to a newer generation of coding schemes which are object-based [3, 4]. These methods rely on the techniques of image analysis and computer graphics where the image signals are represented by their structural features such as contours and regions. In this latter approach, an input video sequence is first segmented into an appropriate set of arbitrarily shaped regions. The features of each region such as shape, motion, and texture/color information are subsequently used in the encoding process. Thus, the success of an object-based method depends largely on the segmentation of the scene (based on its image contents), and its ability to model complex motion more effectively. This object-based approach not only improves the coding efficiency and alleviates blocking artifacts, but it can also support the content-based functionalities mentioned previously by allowing interactivity and manipulation of specific objects within the video stream. These are some of the objectives and issues addressed within the framework of the MPEG 4 and future MPEG 7 standards [5, 6].

In this chapter, we focus our attention on an object-based approach to motion compensated prediction for videophone-type applications. The techniques of image analysis and digital image warping are utilized to form the predicted (current) frame. This is achieved by segmenting the previous frame into a facial region and a number of arbitrarily shaped regions based on the color information with suitable constraints. Each of these regions (represented by an appropriate set of control points), are subsequently transformed to form the predicted image. We first review some of the conventional intraframe and interframe compression schemes along with a brief discussion of several motion estimation techniques. Then, we introduce the methods employed in object-based coding, and more specifically, the mesh-based models that are utilized here. The segmentation of the scene into arbitrarily-shaped regions is subsequently examined using the HSV color space so that the appropriate meshes can be constructed. A discussion of the control point selection algorithm and the assignment of the corresponding motion vectors then follows which concludes the motion compensated prediction process at the encoder. The image warping procedure at the decoder is finally presented and experimental results of various image sequences are shown.
5.2 Intraframe Compression Methods

Data within a static image is compressed by utilizing the statistical and psychovisual redundancies that are present. This process of redundancy reduction is known as source encoding. The diagram in Figure 5.1 helps to illustrate the key elements that may be used in forming a source encoder [2, 67].

![Diagram of the source encoding process]

Figure 5.1: The source encoding process.

The transformation block within the encoder is the first step that may be applied to an input image. This is a one-to-one transformation used to form a new representation of the image which allows for a more efficient compression than the original form. The discrete cosine transform is an effective transformation that manages to accomplish this by packing the energy of the signal into a small number of coefficients. Other mapping functions may include multiresolution techniques such as subband decompositions and the wavelet transform, or numerous other unitary mappings such as the Karhunen-Loeve transformation.

The quantizer in step 2 is optional in the encoding process and is used to generate a limited number of symbols to represent the compressed image. This is a many-to-one mapping which is irreversible, unlike the transformation block. Quantization can be applied to each element of the image in a scalar fashion, or to a number of elements at a time, referred to as vector quantization.

In the final block of the encoder, a symbol coder is used to produce a binary stream of data by assigning a codeword to the symbols that are generated by the quantizer. Variable-length coding (VLC) techniques such as Huffman or Arithmetic
coding can be used to minimize the average length of the binary representation of the signals.

Different compression systems can be realized depending on which blocks are employed in the structure of the encoder. In lossless compression methods, the bit-rate is minimized without any distortion. This is achieved by only using blocks 1 and 3 in Figure 5.1 (both of these blocks are lossless operations) since the quantization step is a lossy process. Lossy compression schemes on the other hand, utilize quantization to minimize the bit-rate in order to attain a given fidelity measure. These methods are briefly outlined next.

5.2.1 Lossless Compression

In certain applications, the compressed images must be decoded without any data loss. This may be the case when dealing with medical images that demand lossless coding techniques for legal purposes. This class of encoding schemes consist of only the transformation, and symbol coding blocks in Figure 5.1, as mentioned earlier. The transformer is used to minimize the entropy of the signal so that high compression can be attained with the symbol coder. Several lossless encoding methods exist, and are capable of achieving compression ratios that range between 2:1 and 6:1.

Lossless predictive coding is a technique used to form an integer prediction of a pixel based on a set of previously encoded neighboring pixels and the entropy coded prediction error. Run-length coding of bit-planes is another common method used in error-free coding. In this approach, the image is decomposed into a set of binary images where the run-lengths in each plane are entropy coded. Another very popular technique is Ziv-Lempel coding which is successfully utilized in compressing binary data files. This is a deterministic method which is based on forming a dictionary of variable length blocks from the input bit string. Each of these blocks are then represented by a fixed-length codeword. The compression ratios that can be reached in each of these techniques depends on the characteristics of the image.
5.2.2 Lossy Compression Techniques

Lossy encoding schemes can provide considerable improvements in data compression over their lossless counterparts. Compression ratios on the order of 25:1 can be achieved with little or no visual degradations in image quality. A compromise in image fidelity is usually made however, for this increase in compression. A lossy compression scheme is formed by including the quantization block of Figure 5.1 into the encoder. A coarser quantization is able to further increase the compression ratio at the expense of a higher signal distortion.

Waveform coding is a class of lossy compression schemes in which the image intensity itself, or some variation of it is encoded [2, 17]. Only blocks 2 and 3 of Figure 5.1 are required in implementing this class of coders, and are thus advantageous from a computational standpoint. The simplest waveform coder available is the basic pulse code modulation, (PCM) scheme where the image intensity is quantized by a uniform scalar quantizer. Other common waveform techniques include delta modulation, (DM) where the difference between two consecutive pixel intensities are coded, and differential pulse code modulation, (DPCM) where the prediction of the current pixel value is determined from more than one previously coded pixel intensity.

Transform coding has proven to be a more effective lossy compression scheme than the waveform coders described above. In transform coding, the image is divided into subimages (i.e. small square blocks) in which a 2D transformation is individually applied to each block [2, 17]. The resulting transform coefficients are subsequently quantized and coded. Thus, all three blocks in Figure 5.1 are utilized in this class of coders. The transformation block allows the encoder to exploit the energy compaction property. This is possible due to the fact that a large fraction of the energy is concentrated in a small fraction of the transform coefficients in the transformed domain. The most effective transformation is one that produces uncorrelated coefficients and packs the maximum amount energy into the smallest number of coefficients. The Karhunen-Loeve transformation, (KLT) is optimal in this sense, however, it is rarely used in practice since it is image dependent, and computationally expensive. The
discrete cosine transform, (DCT) is the most popular transform with good energy compaction properties, relatively low computational complexity, and a fixed set of basis functions (i.e. image independent). The DCT forms the basis of all major international coding standards and is widely available in VLSI hardware for real-time applications.

Once the transform coefficients are available, then one must determine which of these should be retained for coding and how coarsely each should be quantized. This quantization and bit allocation strategy can be performed by: 1) zonal coding in which the coefficients within a specified region are coded, or 2) threshold coding where the transform coefficients are compared with a threshold, and only those above it are coded. The JPEG still image compression standard is an example of a block-based DCT scheme that utilizes threshold coding using a quantization matrix. This matrix is determined according to psychovisual motives and can be scaled to provide various levels of compression.

Despite the popularity of the DCT, there are several drawbacks to this coding scheme and these are evident by the resulting artifacts that appear in the reconstructed images: 1) blurring due to the truncation of the high frequency coefficients, 2) graininess as a result of the coarse quantization of various transform coefficients, and 3) the blocking artifacts due to the independent processing and quantization of neighboring blocks.

A number of other promising lossy image compression methods have been investigated over recent years, including vector quantization (VQ), fractals, subband coding, and wavelets [92, 93, 94, 95]. In vector quantization, an array of samples (pixel intensities or transform coefficients) are quantized into one of a finite number of vector states so that the statistical dependency between data samples is effectively utilized. The success of a VQ technique depends highly on the codebook design (i.e. determining the representative set of finite vector states). The application of fractals to image compression is made possible by exploiting the self-similarity that is found within images. A set of contractive transformations, often affine, are used to map an image into smaller blocks. This allows the regeneration of objects with a high
visual complexity using low information content. In subband coding, an image is decomposed into several nonoverlapping frequency bands where each one is separately encoded according to the statistics and subjective requirements of the respective subband. Low frequency channels may be finely quantized (i.e. major portion of the image energy) while the higher frequency bands can be coarsely represented. Wavelet decomposition is another multiresolution technique that is closely related to transform coding, where the image is expanded onto a set of wavelet basis functions that are well localized in both space and frequency. The interested reader can find further details of the different approaches in the references provided above.

5.2.3 Second Generation Coding Methods

The compression methods discussed thus far all belong to the set of classical coding techniques which attempt to exploit the statistical properties of the information source. Recent efforts have proposed the use of more complicated structural image models and the incorporation of the human visual system into the coding scheme. Techniques that follow this latter approach are referred to as second generation coding methods [96, 97]. This class of coders can be subdivided into two categories: 1) local-operator-based techniques, and 2) contour-texture models. Pyramidal and subband coding methods fall into the first category as they resemble the operation of the human visual system. The contour-texture oriented techniques of the second category attempt to segment the image into textured regions surrounded by contours so that the contours closely correspond to the objects in the scene. The contours and texture information are then coded separately. In [96], two techniques that belong to this category are described, that of region growing, and directional decomposition. Powerful representations are possible with this latter approach which also forms the basis of recent object-based compression schemes. Compression ratios on the order of 100:1 can be achieved with these second generation methods with reasonable quality.
5.3 Interframe Compression Techniques

Interframe coding techniques are used to exploit the temporal redundancies as well as the spatial redundancies that are present within a video sequence. Several different approaches can be used to combine the existing spatial and temporal correlations: 1) 3D waveform coding strategies, 2) motion compensated coding techniques, and 3) analysis-synthesis based methods which rely on more sophisticated scene models. Each of these are discussed below.

5.3.1 3-D Waveform Coding

Three-dimensional waveform coding techniques are a simple extension to the two-dimensional still image compression methods. A number of 3D coding schemes have been investigated, including the 3D DCT, 3D subband and wavelet coders. These methods jointly utilize the spatio-temporal correlation through statistical signal models. The motion information in these 3D coders is not explicit and is handled by adding the time coordinate in the transformation. However, when excessive motion is present between successive frames then the temporal correlation decreases and dramatically reduces the effectiveness of these techniques. In these cases, motion information must be employed in order to increase the interframe correlation.

In three dimensional DCT coding, the video sequence is divided into $N_1 \times N_2 \times N_3$ blocks where $N_1 \times N_2$ represents the spatial dimensions and $N_3$ the temporal dimension of the block. The transform coefficients of each of these volume blocks are quantized and encoded in a similar fashion to the 2D case. The coefficients in most of these blocks will be concentrated in the low spatial and temporal frequency regions allowing many to be truncated in a threshold coding scheme. This 3D transform encoder is advantageous from the point of view that it does not require a separate motion estimation step as mentioned earlier. However, $N_3$ frames must be stored at both, the encoder and decoder and for this reason a small number is usually chosen (i.e. 2 or 4) for practical purposes. Many references for this class of coders may be found in [98].
5.3.2 Motion Compensated Waveform Coding

Motion compensated prediction is a powerful tool that is used to increase the temporal correlation between frames in an image sequence. The task of motion compensation is to estimate the motion between video frames so that the image in the reference frame, $f_r(x_1, x_2, t)$, can be transformed to match or predict the current frame, $f_c(x_1, x_2, t + l\Delta t)$. The error in the prediction can be referred to as the transformed frame difference (TFD) and is expressed by

$$ TFD = \mathcal{D} \{ f_c(x_1, x_2, t + l\Delta t) - T[f_r(x_1, x_2, t)] \} \tag{5.1} $$

where $T$ is a suitable transformation function, $f_r$ and $f_c$ are the continuous time-varying intensity images, $x_1$ and $x_2$ are the continuous spatial coordinates of the image frame, $t$ is the temporal component of the sequence, $\Delta t$ is the frame interval, $l$ is an integer, and $\mathcal{D}$ is an appropriate distance measure that normally takes the form of the $L_2$ or $L_1$ norm. In the simple case where all the elements in the scene are spatially displaced, then a zero-order polynomial transformation, $T_d$, is capable of handling the translatory motion. If the translational model is used to determine the current frame, then the TFD can be referred to as the displaced frame difference (DFD)

$$ DFD = \mathcal{D} \{ f_c(x_1, x_2, t + l\Delta t) - f_r(x_1 + d_1(x_1, x_2), x_2 + d_2(x_1, x_2), t) \} \tag{5.2} $$

where $d_1(x_1, x_2)$ and $d_2(x_1, x_2)$ are the real-valued displacement vectors in the horizontal and vertical directions, respectively, between the times $t$ and $t + l\Delta$. The encoding of the differential signal in equation (5.2) is also known as the temporal prediction error and is usually transmitted to the receiver along with the computed motion vectors. A minimization of this prediction error reduces the bit-rate and also yields a better quality reconstructed image. Thus, motion compensated prediction is a vital component in any video coding environment.

In motion compensated (MC) transform coding, the prediction error is encoded
by dividing the DFD into blocks and coding the DCT coefficients of each block as in the 2D intraframe case. Thus, DCT encoding of the DFD utilizes the spatial redundancy in the prediction error. This coding scheme has been the basis of all major international video standards.

The combination of the spatial 2D intraframe compression methods and the encoding of the motion compensated prediction errors as two separate steps, are referred to as hybrid video coding techniques as opposed to the joint 3D coding schemes illustrated in the previous section. A number of other methods can also be used to encode the prediction error signal such as MC vector quantization or MC subband coding [2].

Motion estimation is an important step that is used to minimize the prediction error. Several different approaches in estimating the displacement vectors, \( d_1(x_1, x_2) \) and \( d_2(x_1, x_2) \) are discussed next.

5.3.3 Motion Estimation

A time-varying image is a two-dimensional projection of a three-dimensional scene. Consequently, the 2D motion that is observed in the image plane is also a projection of the 3D motion. Thus, motion estimation may refer to the motion in the image plane (i.e. 2D motion) or to the motion of the object (i.e. 3D motion). Here, we focus our attention on 2D motion estimation techniques since they are appropriate for our proposed object-based temporal prediction scheme.

The true 2D displacement or motion vectors are projections of the 3D fields onto the image plane and are in general, different from the observable variations of the 2D image intensity pattern. The perceived time-varying pattern is known as the apparent 2D displacement vector or the optical flow. The differences between true and apparent motion may occur as a result of changes in the external illumination (i.e. observed optical flow but no motion), or due to a lack of spatial image gradient (i.e. unobservable motion with no optical flow). Thus, when we speak about the 2D motion field we are referring to the apparent motion since we are only able to observe the optical flow.
The essential problem in 2D motion estimation is to determine the displacement vectors \( \mathbf{d}(x_1, x_2) = [d_1(x_1, x_2), d_2(x_1, x_2)] \) between two frames at times \( t \) and \( t + l\Delta t \). These can be found using forward motion estimation

\[
f_r(x_1, x_2) = f_c(x_1 + d_1(x_1, x_2), x_2 + d_2(x_1, x_2)) \tag{5.3}
\]

or backward motion estimation

\[
f_c(x_1, x_2) = f_r(x_1 + d_1(x_1, x_2), x_2 + d_2(x_1, x_2)) \tag{5.4}
\]

where the temporal subscripts have been dropped for convenience. Backward motion estimation is commonly employed in predictive video compression schemes as it easily facilitates forward motion compensation.

Motion estimation that is based on two frames is an ill-posed problem in the absence of additional assumptions about the nature of the motion. A number of difficulties arise when trying to determine the displacement vectors such as the familiar occlusion problem (i.e. covered/uncovered background), the lack of a unique solution (known as the aperture problem), and the presence of observation noise. A number of approaches have been taken to estimate the motion fields, and these fall into two broad categories: 1) parametric models, and 2) non-parametric models. This first class of models aim to describe the orthographic or perspective projection of the 3D motion using an appropriate representation of the 3D surface. The non-parametric models of the second category apply various smoothness constraints on the 2D motion fields without utilizing the 3D motion models. Optical flow equation based methods, pel-recursive techniques, and block-matching schemes belong to this latter class of models, and are briefly outlined below.

### 5.3.3.1 Optical Flow-based Techniques

The optical flow equation is based on the assumption that the intensity remains constant along a motion trajectory. If \( f(x_1, x_2, t) \) represents the continuous space-
time scalar intensity distribution, then

\[
\frac{df(x_1, x_2, t)}{dt} = 0
\]  (5.5)

describes the rate of change of intensity along the motion trajectory. This total derivative expression can be expanded using the chain rule to yield

\[
\frac{\partial f(x, t)}{\partial x_1} v_1(x, t) + \frac{\partial f(x, t)}{\partial x_2} v_2(x, t) + \frac{\partial f(x, t)}{\partial t} = 0
\]  (5.6)

where \( x = (x_1, x_2) \), \( v_1(x, t) = dx_1/dt \), and \( v_2(x, t) = dx_2/dt \). The expression in equation (5.6) is known as the optical flow equation (OFE) and can alternatively be written as

\[
\langle \nabla f(x, t), v(x, t) \rangle + \frac{\partial f(x, t)}{\partial t} = 0
\]  (5.7)

where \( \nabla \) is the gradient operator, \( \langle \ldots \rangle \) is the vector inner product, and \( v(x, t) = [v_1(x, t), v_2(x, t)] \). However, the 2D velocity field cannot be uniquely determined from the OFE since it consists of one equation in the two unknowns, \( v_1(x, t) \) and \( v_2(x, t) \). Only the component that is in the direction of the spatial image gradient can be determined since the orthogonal component becomes zero due to the dot product. This constraint that the OFE imposes on the component in the direction of the spatial image gradient is consistent with the aperture problem mentioned earlier.

Several techniques have proposed the addition of another constraint so that both components of the velocity vector can be determined. The method of Lucas and Kanada [99] tries to overcome the aperture problem by assuming that the motion vector remains unchanged over a block of pixels by minimizing the error of the OFE over a block of pixels. However, this method is only able to handle a purely translational motion vector uniquely. The method by Horn and Schunck [100] seeks a motion field that minimizes the pixel to pixel variation among the motion vectors. The global smoothness constraint imposed in this technique however, blurs the motion edges at occlusion boundaries. More advanced directional smoothness constraints must be utilized in order to allow sudden discontinuities in the motion field. OFE-based methods
are computationally too demanding for real-time use and their performance also depends on the accuracy of the estimated partials of the image intensity.

5.3.3.2 Pel-recursive Methods

Pixel or pel-recursive motion estimation is a recursive technique that is used to compute the motion vectors on a pixel by pixel basis. The predicted motion vector at each pixel is updated by minimizing the displaced frame difference. The displacement vector, \( d(x_1, x_2) = [d_1(x_1, x_2), d_2(x_1, x_2)] \) is estimated by using various gradient-based optimization techniques to minimize the absolute value of the DFD. A number of numerical optimization techniques such as the steepest-descent, and the Newton-Raphson method, among others have been employed [101]. Initialization of these algorithms is important in order to avoid any local minima. The rate of convergence is also sensitive to the choice of the parameter step size which may affect the output by way of a slow convergence or an oscillatory behavior. Pel-recursive techniques also generate a dense motion field which may substantially increase the bit-rate of the transmitted motion vectors.

5.3.3.3 Block Matching Methods

Block matching techniques are the most popular motion estimators due to their simplicity and practicality. They are widely available in VLSI hardware and are implemented in major international video coding standards such as H.261 and MPEG 1 and 2.

In block matching, images are divided into non-overlapping blocks (i.e. 16 \( \times \) 16 pixels in MPEG 1 and 2 standards) and only one motion vector is estimated for each block. This concept is illustrated in Figure 5.2 for a particular block in the case of backward motion estimation. The discrete displacement vector, \( d = (d_1, d_2) \) of block \( B_c \) in the current frame \( f(n, k + 1) \) is determined by finding the best matching block \( B_p \) in the previous frame \( f(n, k) \) within a search window. The frames, \( f(n, k + 1) \) and \( f(n, k) \) are the discrete scalar intensity images, \( n = (n_1, n_2) \) are the discrete spatial coordinates, and \( k \) is the discrete temporal component.
Figure 5.2: Block matching in backward motion estimation.

Various matching criteria can be utilized to quantify the match between two blocks. The mean square error (MSE) is a common criterion that is typically employed

\[
\text{MSE}(d_1, d_2) = \frac{1}{B_1 B_2} \sum_{(n_1, n_2) \in B} [f(n_1, n_2, k) - f(n_1 + d_1, n_2 + d_2, k + 1)]^2
\]

(5.8)

where \( B \) represents a block of size \( B_1 \times B_2 \), and \( d = (d_1, d_2) \) are the discrete candidate displacement vectors. The MSE measure can also be formulated for color or vector images as follows

\[
\text{MSE}(d_1, d_2) = \frac{1}{B_1 B_2} \sum_{(n_1, n_2) \in B} \| f(n_1, n_2, k) - f(n_1 + d_1, n_2 + d_2, k + 1) \|^2
\]

(5.9)

where \( f \) denotes the vector color image, and \( \| \ldots \| \) is the Euclidean distance measure. The estimate of the displacement vector is the value of \( (d_1, d_2) \) which minimizes
the MSE. By expanding the DFD into a first order Taylor series, it can easily be seen that the minimization of the MSE is equivalent to the minimization of the OFE in the Horn and Schunck method mentioned earlier.

The mean absolute distance (MAD) measure is another criterion that is used primarily for VLSI implementations in order to reduce the number of computations performed. In this case, the square operation in expression (5.8) is replaced by the absolute operator.

A number of search strategies can be used to find the best matching block between two frames. The full search technique evaluates the matching criterion exhaustively for all values of \( d = (d_1, d_2) \) within the defined search window. Faster search algorithms such as the three-step search, and the cross search can also be employed [102]. However, these techniques are suboptimal since the distortion measure is minimized only over a predetermined subset of pixel locations.

### 5.3.3.4 Deformable Block Matching

Deformable or generalized block matching has been proposed in order to overcome the limitations associated with the conventional block matching techniques (i.e. translational models). In deformable block matching, the current frame is divided into triangular, rectangular, or arbitrary quadrilaterals and a search is made for the best matching triangle or quadrilateral in the previous frame (i.e. for backward motion estimation). A spatial transformation is subsequently used to describe the motion between the patch in the current frame and its best match in the previous frame. Figure 5.3 illustrates this case. This approach outperforms the translational block matching schemes by allowing for more complex motion types such as rotation and zooming to take place [103].

The selection of different patch shapes directly affects the spatial transformation that must be used in order to model the geometrical mapping. The affine transform offers six degrees of freedom (i.e. six independent parameters) and is well suited for triangle-to-triangle mappings. The coordinates of two pixels are related by the
following transformation in the affine case

\[ x_{1c} = a_1 x_{1p} + a_2 x_{2p} + a_3 \]  

\[ x_{2c} = a_4 x_{1p} + a_5 x_{2p} + a_6 \]

where \( x_{1p} \) and \( x_{2p} \) represent the horizontal and vertical continuous spatial coordinates of a pixel in the previous frame, respectively; \( x_{1c} \) and \( x_{2c} \) the horizontal and vertical continuous spatial coordinates of the corresponding transformed pixel in the current frame, respectively; and \( a_i \), for \( i = 1, \ldots, 6 \) the coefficients of the affine transformation. Three points or pixels (i.e. triangle) are sufficient to uniquely determine the mapping coefficients. Other spatial transformations include the perspective and bilinear mappings which have eight independent coefficients and are suitable for use with rectangular or quadrilateral patches [104]. The conventional block matching
scheme is actually a special case of the affine transform with \( a_1 = a_2 = a_4 = a_5 = 0 \) which results in the translatory model.

The superior motion tracking and rendition of deformable block matching however, comes at an increased cost in computational load. In the generalized case, the best matching quadrilateral in the previous frame must be determined (within a search window) for the rectangular block in the current frame. For each possible quadrilateral, the coefficients of the specified spatial transformation (i.e. perspective or bilinear for the example in Figure 5.3) must be calculated using the coordinates of the four matching corners (i.e. between the rectangular block, \( B_c \) and the quadrilateral block, \( B_p \)). The points within the quadrilateral are transformed using the computed mapping coefficients, and the MSE is finally calculated. In the full search method this would require going through this process for every possible quadrilateral within the search window. For an \( N_s \times N_s \) search area, this would require \((N_s + 1)^4\) different mappings and MSE calculations versus \((N_s + 1)^2\) in the conventional case. Several techniques have been proposed, however, to reduce the number of calculations by using faster search algorithms.

In deformable block matching, the motion within a triangle or quadrilateral is tracked by utilizing the pixel correspondences that are determined at the corners of the patch (i.e. matching of patch corners in the current frame to those in the previous frame). Thus, it is important that the patches that are formed contain only one moving object, otherwise, a single spatial transformation will not be sufficient to model the motion within the patch. A solution to this, is to partition the patches according to the content in the scene. Adaptive or content-based mesh models are suitable in handling such cases, and these are employed in our proposed object-based prediction scheme, described later in the chapter.

5.3.4 Analysis-Synthesis Coding

A new class of motion compensated coding techniques, known as analysis-synthesis coders, have been proposed recently. These encoders are based on a set of structural image models rather than the statistical signal models employed in the block-based
methods. Analysis-synthesis coders have shown considerable improvements over the conventional schemes and are better suited for low bit-rate applications. A structural representation of the image also allows for content-based functionalities within the system such as interactivity, content-based storage/retrieval, and editing/compositing.

An analysis/synthesis coder is comprised of three essential steps: 1) image analysis, 2) image synthesis, and 3) parameter coding. In the analysis step, the current frame is segmented into a set of shape and motion parameters. In image synthesis, the current frame is reconstructed using the estimated shape and motion parameters as well as the previously encoded frame. The difference between the actual current frame and the synthesized one provides a measure of model compliance. The shape, motion, and color (for model failure regions) parameters are finally entropy encoded for transmission in the third step.

Analysis-synthesis coders include a number of different approaches such as the object-based coding methods, and the knowledge/semantic based coding techniques. In object-based coding, a scene is represented in terms of arbitrary, moving objects where each object is described by a predetermined source model [3, 105]. In knowledge-based coding, a priori information about the content of the scene is utilized in order to design generic wireframe models for the objects of interest (i.e. facial models for videoconferencing applications) [106, 107]. The 3D motion and structure (depth) must be estimated at the encoder so that the global motion of the wireframe and its changing structure are tracked. Semantic coding is used in conjunction with the global knowledge-based methods in order to model the local motion (i.e. facial deformations) in terms of a set of facial action units. However, many difficulties arise in the knowledge/semantic based coders which include: 1) adaptation of the generic wireframe to the actual speaker, 2) estimation of the 3D global motion and structure parameters (i.e. unknown depth), 3) characterization of the local motion, and 4) the generality of the coding algorithm.

The motion compensated prediction scheme we propose belongs to the object-based coders above and is investigated next.
5.4 Object-based Coding

The conventional block-based schemes do not adequately describe the motion fields due to the limitations of the translational motion model as mentioned earlier. The use of rectangular blocks does not allow for complex motion types that objects typically undergo within a scene (i.e. zooming, rotation, different motion within a given block). Object-based methods have recently been proposed to improve upon the deficiencies of the MC/DCT compression schemes by devising more realistic 2D motion field models. The general object-based techniques can be categorized into two different approaches: 1) arbitrary 2D or 3D object models with 2D or 3D motion, and 2) mesh-based models that utilize spatial transformations.

In the first approach, each scene is segmented into individual moving objects that are based on different source models. Each moving object is subsequently described by three sets of parameters defining its shape, motion, and color. The source model of a rigid 2D planar object with 3D motion can be represented by a six parameter affine or eight parameter perspective model [3]. These parameters are used to describe the motion and position of the rigid 2D object and are referred to as the mapping parameters. The frame difference between two successive frames is used to set up a vector matrix equation as a function of the mapping parameters and the weighted local gradients. This system is linearized and the parameters are estimated by a minimization of the mean square displaced frame difference. The simultaneous estimation of the model parameters and the scene segmentation in this approach, however, is computationally demanding. For this reason, the source model of 2D flexible objects with 2D motion was introduced [4, 108]. This also belongs to the first approach of the general object-based methods. In this scheme, the image is partitioned into three regions: 1) unchanged areas, 2) moving areas that are model compliant, and 3) regions that are model failures. Motion estimation is performed using a hierarchical block-matching scheme in the changed regions (i.e. using a change detection mask) [109]. Model compliant moving objects are encoded by the 2D motion vectors and their respective shape parameters (i.e. polygon-spline approximations). The model
failure areas are regions that cannot adequately be described by motion parameters and so these are encoded by shape and color information.

The second approach of object-based methods relies on a set of simpler motion compensation schemes that utilize spatial transformations on a predetermined partition of the image into triangular or rectangular patches. Our proposed motion compensated prediction technique belongs to these mesh-based models and are outlined next.

5.4.1 Mesh-based Modeling

Motion compensated prediction schemes that are based on spatial transformations have recently been applied to the video compression problem [88, 91, 90, 87]. The concept behind this approach is to partition the current frame to be coded, into triangular or rectangular patches that define a 2D mesh. Motion compensation is performed by spatially transforming or digitally warping each mesh-patch according to a suitable mapping function. The results obtained from these techniques have proven to be superior to those of the conventional block-based MC/DCT approaches.

The earliest form of mesh-based motion compensation was proposed by [88], known as control grid interpolation (CGI) in their work. In this scheme, the current frame is partitioned into square patches where the four corners are referred to as control points. For each of these square blocks in the current frame, a best matching quadrilateral is determined in the previous frame. This is accomplished by finding the best matching spatial displacements for each of the control points in the current frame through an iterative procedure. The remaining points are found by interpolating between the control point displacements.

Nakaya and Harashima [91] have proposed a generalized approach to using spatial transformations based on a predetermined partition of the image into regular triangular or rectangular patches. Their findings suggest that the affine transformation with triangular patches (using a backward motion compensation scheme) outperforms the bilinear transformation with rectangular patches. In the affine case, the image is segmented into triangular patches, where the control or grid points are the vertices
of each triangle. The motion vectors of these grid points in the current frame are estimated by determining the corresponding grid points in the previous frame. The affine transformation is then used to map the texture within the warped triangles of the previous frame to their respective triangles in the current frame. The motion vectors of the grid points are estimated using an iterative hexagonal matching technique. This is done in a two step process where the motion vectors are first approximated using conventional block matching, and subsequently refined using an iterative local minimization of the prediction error. The computational cost of this method is quite expensive and thus, a faster algorithm (with a slightly inferior performance) is also suggested. Ideally, the motion estimation algorithm should determine the motion vectors that minimize the global approximation error over the whole image rather than the local approach taken. However, this is mathematically intractable and so the iterative grid point motion vector approach serves as a good approximation.

Motion compensation using the regular rectangular (i.e. square) mesh structure proposed in [91] is illustrated in Figure 5.4. The image in the current frame is partitioned into regular square patches, and the corresponding best matched deformed patches are shown in the previous frame. In this example, 25 grid points (i.e. motion vectors) are used to model the complex motion (zooming and rotation) that takes place in the scene. In general, smaller patch sizes can represent the content in the scene more effectively, and as a result, lead to more accurate motion fields. The smaller patches, however, correspond to a greater number of grid points, meaning that a larger number of motion vectors must be transmitted. This can be critical in a low bitrate environment. In addition to this, the smaller squares may result in degenerate rectangles through the matching process. This occurs when the angle between two sides of a polygonal element becomes obtuse. The problems associated with larger mesh patches are also evident however, when objects undergoing different motion fall within the same patch. This can lead to geometrical deformations in the reconstructed image. Nevertheless, these types of artifacts are more acceptable to a human observer than the blocking artifacts of the MC/DCT schemes.

Several efforts have been made to improve upon the techniques that utilize uniform
or regular meshes. In [90], a deformable mesh structure using nonuniform samples has been proposed. The nodal points in the mesh are more densely distributed in regions containing features such as edges and corners. The mesh is generated through an energy minimization procedure of a weighted sum of several energy functions. A gradient descent algorithm is used in an iterative process with a regular mesh as the initial condition. A similar gradient-based minimization is used to track the nodal displacements using the mesh of the previous frame as the initial solution. The complexity of this approach, however, is quite high and is also susceptible to the problems of local minima and slow convergence rates. The authors suggest the use of other non-iterative solutions for real-time applications. Motion compensation using an adaptive patch structure has also been proposed in [87]. This scheme utilizes arbitrary quadrilateral patches in a forward matching scheme. The previous frame is initially partitioned into square patches, and is subsequently modified to adapt to the features of the image. This is done by moving each grid point to its closest edge point within a local neighborhood using an edge detector. Motion estimation of the control points is determined using a modified block matching technique suitable for real-time
implementation. Both of the techniques just described have shown improvements over the regular mesh structures, however, a more accurate representation of the motion can be obtained by using a content-based mesh design procedure. This will not only improve the motion estimation, but it will also allow for a true object-based representation (based on the scene content) that will allow for future content-based functionalities in digital video. The authors in [90, 87] have also suggested that future mesh-based representations be designed according to a segmentation of the scene. The motion compensation scheme that we propose in our work employs a content-based 2D mesh that is designed using a color segmentation method in the HSV color space. Figure 5.5 illustrates a mesh design for an image that contains two regions against a uniform background. This is a simplified scenario of a typical videoconferencing

Figure 5.5: A content-based 2D mesh design for motion compensation.
sequence of one person against a stationary background. Each region is represented by a suitable set of triangular mesh patches that models the underlying motion. This type of mesh design allows for better motion estimation, smaller number of motion vectors (i.e. grid points), and a powerful representation of each region. This latter property allows us to generate higher level descriptions of the contents within the scene, that is, region 1 and region 2 may belong to object 1 (i.e. human), and region 3 may belong to object 2 (i.e. background). This can facilitate interactivity, editing functions (i.e. cut and paste, merging of natural and synthetic imagery), and content-based retrieval from video databases or the Internet (i.e. retrieve a newscaster with black hair, blue eyes, against a stationary red background). Thus, the first critical step involved with this approach is image segmentation. A color segmentation scheme is utilized here that operates in the HSV color space. This is outlined next.

5.4.2 Color Image Segmentation

The segmentation process for isolating the skin colored clusters within an image was outlined in the detail in the previous chapter. This allowed us to automatically extract the facial regions by utilizing the apriori knowledge of the skin-tone distributions in the HSV color space. Having determined the facial regions, then the remaining part of the image must be segmented into a set of arbitrarily-shaped uniform regions. For this purpose we employ a recursive 1D histogram thresholding procedure, once again, in the HSV color space. The proposed technique is robust, suitable for real-time implementation (i.e. due to the 1D histogram approach), and very intuitive in describing the color/intensity content of a region.

The Hue component in the HSV color model can be effectively employed to segment the color content within a scene. However, as mentioned in the previous chapter, the Hue attribute is ineffective and unreliable when the Saturation or Value components are low. Therefore, we partition the image into the following three primary regions so that an appropriate segmentation scheme can be applied within each region: 1) an achromatic, 2) a chromatic, and 3) a transitional area. The achromatic regions are characterized by low values of Saturation and Value, and consist of the
black, white, and gray areas within the scene. Threshold values of $S \leq 10\%$ and $V \leq 20\%$ were used to define the achromatic sector of the HSV space. A similar Saturation threshold has been selected in [76] to partition the achromatic sector of the HVC space without enforcing an intensity restriction. The intensity information, however, is important, [61] and erroneous results may be obtained if this latter restriction is not imposed [110]. The Value component (i.e. the brightness) is used to segment the achromatic regions of the image. The chromatic region (region 2), on the other hand, is described by high values of Saturation and Value where the Hue has great discriminating power and can be effectively used to segment the chromatic parts of the image. Threshold values of $S \geq 20\%$ and $V > 20\%$ were selected in defining this second region. Finally, the third region separates the chromatic and achromatic areas, and is referred to as the transition region. Thresholds of $10\% < S < 20\%$, and $V > 20\%$ were chosen for this latter region. Slices of this solid correspond to annular rings in the HSV model. The Hue component in this transition region is once again unreliable and this can also be observed visually from Figure 4.6. Pixel values within this region have very little chroma, and thus, are better characterized by the Value component. This partitioning of the HSV hexcone model into the three primary regions is summarized in Table 5.1 below. A simple two-region model has also been

<table>
<thead>
<tr>
<th>Region</th>
<th>Bounding Thresholds</th>
<th>Segmentation Cue</th>
</tr>
</thead>
<tbody>
<tr>
<td>Achromatic</td>
<td>$S \leq 10%$</td>
<td>$V \leq 20%$</td>
</tr>
<tr>
<td>Transitional</td>
<td>$10% &lt; S &lt; 20%$</td>
<td>$V &gt; 20%$</td>
</tr>
<tr>
<td>Chromatic</td>
<td>$S \geq 20%$</td>
<td>$V &gt; 20%$</td>
</tr>
</tbody>
</table>

Table 5.1: **Partitioning of the HSV hexcone model.**

proposed for segmentation purposes in the similar HSI space [111]. In this scheme, the original image is split into only two regions (chromatic and achromatic) by using the average value of the peaks found in the Saturation histogram, as a threshold value. There are two problems associated with this approach: (i) threshold values may be over or under estimated due to the averaging process which may result in an incorrect partition of the chromatic and achromatic regions, and (ii) no intensity information
is taken into account, which may lead to erroneous results due to the low intensity value pixels.

Once the image has been partitioned into the three primary regions above, then a histogram thresholding procedure is carried out within each region using the appropriate cue.

5.4.2.1 Histogram Thresholding

Segmentation within the achromatic region is performed by using the histogram of the Value component. The Value histogram is first formed and smoothened by the scale-space filtering approach described in the previous chapter. The largest peak is then selected and the valleys are subsequently found on either side of this peak. Pixel values within the two valley points are classified as a uniform area. A set of binary operations which include median filtering, and region removal are used to remove isolated pixels, and small regions (i.e. less than a predefined threshold), respectively. This process is repeated recursively until all the pixels within the achromatic region are segmented into significant areas of uniformity (i.e. no more regions can be further extracted from the histogram after the small region removal step).

The procedure just described is also carried out using the Value histogram of the pixels within the transitional region. Areas within this region appear to have some chroma component and, therefore are kept disjoint from the achromatic region.

Finally, the chromatic region is segmented by using the Hue histogram of the chroma pixels as defined in Table 5.1. However, we have found that subdividing the chromatic area further into subregions yields an improvement in the segmentation results. This division is carried out at the valleys (i.e. between peaks) of the smoothened Saturation histogram of the chromatic region. In effect, this partitions the chromatic areas into varying levels of saturation for improved results (i.e. two areas with the same Hue but different Saturation values are not grouped together). Segmentation is performed within each of these chromatic subregions by using the histogram of the Hue component (i.e. as done with the Value component above).
5.4.2.2 Post-Processing and Region Merging

The recursive histogram procedure described in the previous section is applied to each of the three primary regions, until no areas of uniformity can be further extracted. However, a number of pixels will still remain unclassified as a result of this process (i.e. due to small region removal, median filtering, etc.). These pixels are subsequently combined into the best matching region (within a spatially local window) from the set of regions obtained in the initial histogram extraction process as follows. The image is progressively scanned (in a raster scan fashion) and a $3 \times 3$ window is formed for each unclassified pixel that borders at least one pixel (in the 8th connected nearest neighbor sense) from an initially segmented area. The $L_2$ norm is computed for each pixel in the window, with respect to the central pixel (i.e. the unclassified pixel). The smallest value is taken and compared to a predefined threshold. If it is less than the threshold value, then the central pixel is incorporated into the area that the corresponding pixel (i.e. the one with the smallest $L_2$ norm) belongs to. If it exceeds the threshold value, then the central pixel is left unclassified. Pixel sites are revisited through a number of iterations until all the unclassified pixels are grouped to an appropriate region. When no groupings are made within a particular iteration, then the threshold values are increased so that the process converges. The selection of the initial threshold value is quite small, and is gradually relaxed (i.e. increased) until all pixels are classified. This process is very fast since there are usually a small number of pixels (typically at the borders of regions) requiring few iterations.

Once all of the pixels have been classified, then a series of binary morphological operations are used to refine the extracted regions. A binary morphological opening operation is first used to remove small spurs and thin channels, followed by a binary morphological closing operation to fill in small holes and gaps. The description of these operators can be found in the previous chapter.

At this stage, the segmentation of the image into a set of refined, uniform regions is complete. However, an oversegmented region may result if the threshold for small region removal is set too low. Region merging is used to overcome this situation.
by joining bordering regions with a similar average Hue value. Adjacent regions are merged if the Euclidean distance of the average RGB values of two regions is less than a set threshold. Region merging is performed in the RGB space due to the lack of an appropriate distance metric in the HSV color space. Regions can be merged so that the smallest region is of some minimum size, or that a specific number of regions are obtained. Here, we select a fixed threshold based on experimental values to reduce the computational complexity. Setting an appropriate threshold can also reduce the regions so that they coincide with semantically meaningful objects.

5.4.3 Content-Based Mesh Design

In a content-based mesh design, the patch boundaries are designed to fit the scene content and motion edges. This type of non-uniform mesh structure allows for a more accurate motion estimation and compensation, as mentioned previously. Several approaches have been proposed for adaptive mesh design: (i) the optimization method of [90], already mentioned in Section 5.4.1, (ii) the split and merge technique in [112], and (iii) the spatio-temporal gradients method [112]. The split and merge method aims to design a locally optimal mesh structure in the sense of minimizing the displaced frame difference (DFD), and is initialized with a uniform mesh. In the spatio-temporal gradient scheme, the image is partitioned into triangles in such a way that a predefined measure of the DFD within each patch attains approximately the same value. The algorithm in this latter scheme works by iteratively selecting a pixel location with a high intensity gradient magnitude inside the region of interest and assigning a node to that location. A region is subsequently grown around the node location and all the pixels inside that region are marked so as to prevent them from being picked as a node location in the future. The process continues until a desired number of nodes are reached or until all locations in the region of interest are marked. After all nodes are assigned a location, then the triangular meshes are finally constructed using the Delaunay triangulation method [113]. In both of the two latter techniques, (ii) and (iii), a dense 2D motion field estimation is necessary which also increases the algorithmic complexity.
The essential idea in all three of the above approaches is to select the nodal points at the edges and features of the image. However, in our case, we already have a set of uniform regions due to the color segmentation process. Thus, we select the nodal (i.e. control) points along the edges of each of the extracted regions. We also select an additional nodal point at the center of mass of each region. The procedure for generating the triangles within each region is as follows. First, the boundary for the region of interest is extracted. A set of nodal points are subsequently placed along the boundary at evenly spaced intervals (i.e. every n pixels) or according to the curvature or bending energy of the boundary [67]. This latter approach is a better representation, as it allows for a more effective placement of the nodal points (i.e. more points are allocated at the boundary segments with a higher curvature). We utilize the first approach as it serves as a good approximation at a reduced complexity. Next, the center of mass (COM) of the region is determined. The triangles are finally constructed by connecting each boundary nodal point to the COM, and each boundary point to its counter clockwise (CCW) neighboring boundary point. This process must be consistent at both, the encoder and decoder, so that the triangles are formed in a consistent manner. Thus, we select the top-left-most point as the initial boundary point, and proceed in a CCW fashion around the boundary. The results from this procedure are illustrated in Figure 5.6 for region, R₁. In this example, the region is partitioned into 9 triangles (Tᵢ for i = 1, ..., 9) using 10 nodal points (9 boundary points, Bᵢ, for i = 1, ..., 9, and 1 COM point, C₁). In the case that the shape of a region prohibits a boundary point from joining the COM (i.e. the connecting line crosses into another region), then the region is split into two sub-regions by joining the two boundary points that neighbor the unconnected boundary points and which are also connected to the COM. Figure 5.7 illustrates this particular case. The point C₁ is the center of mass of the entire region R₁. However, C₁ cannot connect to points B₇, B₈, and B₉, so two sub-regions are formed by joining B₆ and B₁₀. As a result, sub-region, R₁₁ consists of points (B₁, B₂, B₃, B₄, B₅, B₆, B₁₀, B₁₁, B₁₂, B₁₃, C₁) and sub-region, R₁₂ of points (B₆, B₇, B₈, B₉, B₁₀, C₂). The procedure just described is carried out until all the boundary points are triangulated with an
Figure 5.6: Nodal point selection and triangulation for a segmented region when all boundary points connect to the center of mass, \( C_1 \).

Figure 5.7: Nodal point selection and triangulation for a segmented region when several boundary points cannot connect to the center of mass, \( C_1 \). In this case the region \( R_1 \) is split into two sub-regions, \( R_{11} \), comprised of \((B_1, B_2, B_3, B_4, B_5, B_6, B_{10}, B_{11}, B_{12}, B_{13}, C_1)\), and \( R_{12} \) consisting of \((B_6, B_7, B_8, B_9, B_{10}, C_2)\).
appropriate COM (i.e. certain shapes may require the subdivision of a region into more than two sub-regions).

Finally, we bring up one last case that may arise, that is, when the COM falls outside the region under consideration. In this situation, we divide the region into two sub-regions either horizontally or vertically depending on the bounding rectangle of the region (i.e. formed at an orientation of $0^\circ$). If the horizontal dimensions of the bounding rectangle are greater than the vertical dimensions of the bounding rectangle, then we partition the region vertically at the horizontal midpoint of the rectangle. If, on the other hand, the vertical dimensions are greater than the horizontal ones, then we divide the region with a horizontal line at the vertical midpoint of the bounding rectangle. The example in Figure 5.8 illustrates the former case. The region $R_1$, is divided into two sub-regions since the COM (i.e. $C_1$) falls outside the region. The region is divided vertically since the horizontal dimensions of the bounding rectangle are greater than the vertical ones. Once the sub-regions are formed, then the nodal point placement, COM determination, and triangulation is carried out in the usual manner.

The triangulation process using the COM approach we have outlined above, was
selected in order to minimize the number of degenerate triangles obtained when temporarily tracking the nodal points. This scheme increases the number of control points within each region by one (i.e. or equal to the number of sub-regions within a region), however, less refinement of the mesh is required to preserve its connectivity. Other nodal placement strategies and triangulation techniques (i.e. Delaunay method with suitable constraints on the triangular shapes) are currently being explored to reduce the number of necessary control points, and at the same time handle large mesh deformations.

Having outlined the nodal placement and triangulation scheme for a single arbitrarily-shaped region, we next illustrate the implementation of this approach over an entire image or video frame which consists of a set of regions. Figure 5.9 a) depicts a scene that is partitioned into four disjoints regions, $R_i$, for $i = 1, \ldots, 4$ obtained through the segmentation process. The image is scanned in a top-to-bottom and left-to-right fashion in order to determine the sequence in which the regions are to be processed. Region, $R_1$ is encountered first, and therefore the nodal point placement and trian-
gulation is carried out for this region first. The COM is determined (i.e. $C_1$ shown in part b) of the figure), and the boundary is extracted so that the boundary nodal points can be determined. The boundary point placement scheme begins at the top left-most point of the region boundary. Nodal points are inserted at every $n$ pixels as described earlier. If the boundary of the region coincides with the edges of the image, then points are only placed at the corners of these segments (i.e. $B_1$ for region $R_1$), and where the region borders one or more other regions (i.e. $B_2$ and $B_8$ for region $R_1$). Additional points are also placed where the region neighbors two or more other regions (i.e. points $B_4$, and $B_6$ for the region $R_1$). Once the nodal points have been determined, then the mesh for the region is constructed by the triangulation method described earlier. The pixels within the region are finally marked so that the next region may be obtained and processed. In Figure 5.9 a), the next region corresponds to $R_2$. The procedure just described is repeated successively for each of the regions until the complete mesh is generated. The wireframe in Figure 5.9 b) illustrates the results of the proposed mesh generation process. Only the boundary points for region $R_1$ have been labeled in order to keep the diagram legible. We also note that nodal points do not have to be determined for region boundaries that border marked pixels. The control points of the neighboring region can be utilized since they have already been determined (i.e. $B_6$, $B_7$, and $B_8$ for region $R_2$). In Figure 5.9, we have illustrated four arbitrary regions and generated the appropriate mesh within each region. A mesh, however, is not constructed within the region that corresponds to the facial region. We have found that warping the facial regions results in visible distortions of the facial features (i.e. since the facial area is not uniform and the appropriate placement of nodal points along the facial features is a difficult task due to small size, occlusion, etc.). Thus, we track the facial area and employ intra or interframe coding techniques for this particular region.

The mesh design illustrated in Figure 5.5 is an alternative nodal placement and triangulation scheme that may also be utilized in cases where an object is moving against a uniform background (i.e. videophone type application). However, the procedure outlined above handles complex backgrounds where the objects may also appear
at the edges of the image. We also note, that if the wireframe design in Figure 5.5 is utilized, then a mesh is not constructed within the region that corresponds to the facial area (i.e. region 1) for the same reasons cited above.

5.4.4 Mesh Tracking

Having constructed the mesh structure for one frame, then we must track its temporal evolution over a sequence of frames. For each nodal point of the mesh, a motion vector is found by estimating the forward motion between the previous and current frames. These motion vectors are used to warp the mesh and its contents in the previous frame to that in the current frame. In this way, the mesh is continuously propagated for each predicted frame.

The change in the shape of the mesh over time is determined by the displacement of the nodal points. Thus, we must establish a correspondence between the nodal points in the previous and current frames. Reasonable tracking performance has been reported in [114], using a standard block matching technique (16×16) and subsequent motion vector refinement in their object-based mesh motion estimation scheme. In our proposed method, we use a modified block matching technique similar to [115], to estimate the displacement of each point independently. This scheme has been shown to provide far better estimates of the motion vectors, with a computational complexity comparable to that of standard block matching [115, 87, 116]. In this modified scheme, the motion of each nodal point is estimated by using a square block centered around it, and by subsequently applying a block matching technique. Figure 5.10 illustrates the structure of the block utilized in our nodal point matching process. A larger block size of 21×21 has been chosen to give better results in areas with little detail or with spatially periodic textures. The weights in the figure (i.e. $w_{i,m} = 2, 3, \text{ and } 4$) were empirically selected so that an error in one of the center pixels of the block has a higher cost than an error in one of the outer pixels (i.e. since we are looking for an accurate match of the central pixel and not of the block). The technique in [115] uses the mean absolute difference as an error criterion and a quincunx subsampling of the outer part of the block shown in Figure 5.10. In our
scheme, we use the complete block structure (i.e. as shown in the figure), and the mean square error (MSE) criterion for vector valued color signals. The MSE measure (i.e. Euclidean distance) is formulated by applying the mask above to each of the RGB channels within the search window as follows

\[
MSE(d) = \frac{1}{B} \sum_{l=-10}^{10} \sum_{m=-10}^{10} w_{i,m} \| f(p_{i} + (l, m), k + 1) - f(p_{i} + d + (l, m), k) \| \quad (5.12)
\]

where \( p_{i} = (p_{i1}, p_{i2}) \) are the coordinates of the \( i^{th} \) nodal point, \( d = (d_1, d_2) \) are the candidate displacement vectors, \( w_{i,m} \) are the weights in the block structure, and \( B \) is the number of pixels in the mask (i.e. \( B = 21 \times 21 = 441 \)). The motion vector for the nodal point \( p_{i} \) is determined by selecting the value of \( d \) which minimizes the value of the MSE above

\[
MSE(d_{p_{i}}) = Min |MSE(d)| \quad \forall \{ d \in W_{p_{i}} \} \quad (5.13)
\]

where \( d_{p_{i}} \) is the motion vector for nodal point \( p_{i} \), and \( W_{p_{i}} \) is the search window for
We have found that the added complexity of using the color information and the full block structure produces better matching results without requiring any further motion vector refinement. In certain cases, however, the estimation of the nodal point motion vectors can lead to potential problems. This may occur in smooth or spatially periodic image areas where the MSE of a long motion vector may be slightly smaller than the correct shorter one. In this situation, the longer motion vector would be erroneously selected according to (5.13) resulting in a geometrically distorted image. Clearly, the longer motion vectors must only be selected if the MSE measure is significantly lower. Thus, in order to alleviate these problems, we modify the MSE measure in (5.12) as follows

\[ E(d) = k \cdot \text{MSE}(d) \quad \text{if} \quad \{ |d_1| + |d_2| > c \} \quad (5.14) \]
\[ E(d) = \text{MSE}(d) \quad \text{otherwise} \quad (5.15) \]

while the corresponding motion vector selection scheme of (5.13) is reformulated as follows

\[ E(d_{p_i}) = \text{Min}\{|E(d)| : d \in W_{p_i}\} \quad (5.16) \]

where \( E(\cdot) \) is the penalty function, \( k \) is the penalty factor, and \( c \) is a predefined distance threshold (the coefficient values for \( c \) and \( k \) were experimentally determined). In [115], an error function with a distance penalty factor (using suitably determined thresholds) has also been selected. We also note that the nodal points that lie on the corners and edges of the previous frame are constrained to lie on the corners and edges of the predicted image, respectively. This ensures that the mapping preserves the shape and size of the transformed image.

The complexity of the tracking problem increases when multiple objects within the scene undergo arbitrary motion due to the covering and uncovering of scene objects. Presently, we do not take into account these latter two cases, however, we outline below how our proposed method can be extended to handle these cases. In our scheme, we do however take into account the covering of scene objects by the facial region. For the videophone-type applications considered here, we assume that
the facial region is always a foreground object (i.e. is not occluded).

Figure 5.11 demonstrates how the point matching process is modified around the facial region in order to minimize potential distortions. Part (a) of the figure depicts a facial region, \( R_f \), in frame \( k \), which moves horizontally towards the left part of the image in the subsequent (i.e. predicted) frame, \( k+1 \). The facial region, in this case, is in the foreground of two stationary regions, \( R_1 \), and \( R_2 \). In the normal point
matching procedure, the best match for nodal point \( p_5 \) may be \( p_5' \) shown in part (b) which leads to a slight distortion of the background. Thus, to overcome this, we carry out the following procedure: (1) The normal point matching process is first applied as in part (b) of the figure. (2) The nodal points which are covered by the facial region in the predicted frame, \( k+1 \) (i.e. \( p_4 \) and \( p_5 \) in our particular example) are assigned the motion vectors of their nearest neighbor nodal points which are not covered. In the example of Figure 5.11, nodal point, \( p_5 \) remains stationary (since its nearest uncovered neighbor, \( p_1 \) is stationary) and \( p_4 \) also remains stationary (i.e. since \( p_3 \) is stationary). The predicted frame is reconstructed (i.e. using the warping technique described in the next section) and the facial region is overlaid over the reconstructed image. This scenario is illustrated in part (c) of the figure. (3) The MSE is computed within the regions, \( R_1 \) and \( R_2 \) for both cases (i.e. part (b) and (c) ) and the selection of the motion vectors for the covered nodal points (i.e. \( p_4 \) and \( p_5 \) ) is made according to the smallest MSE case. In Figure 5.11, this would correspond to part (c) of the figure which also minimizes the visible distortions. We note that nodal points on the boundary of the facial region which do not get covered by the facial area in the predicted frame are constrained to lie on the boundary during the matching process. This is the case, since uncovered regions are not taken into account in the present scheme.

Finally, we outline how our proposed method can be extended to handle uncovered scene objects. Figure 5.12 (a) depicts two regions, \( R_1 \), and \( R_2 \) which undergo complex motion. The nodal point, \( p_1 \) in the previous frame, \( k \) moves to position, \( p_1' \) in the predicted frame, \( k+1 \) as shown in part (b) of the figure. This situation may correspond to one of the following two scenarios: (i) region, \( R_1 \) is in the foreground of \( R_2 \), in which case \( R_1 \) uncovers either a new scene object, \( R_3 \) and/or the existing region \( R_2 \), or (ii) region, \( R_2 \) is in the foreground of \( R_1 \) in which case the exposed region, \( R_3 \) corresponds to the existing region \( R_2 \) (i.e. due to the complex motion) or to a newly appearing scene object. If in either of the two cases the region, \( R_3 \) corresponds to a newly appearing object, then our warping prediction scheme will not reflect these particular scene changes. However, this can be handled by identifying the regions that
Figure 5.12: (a) Two regions, $R_1$, and $R_2$ which undergo complex motion within the scene, and (b) region, $R_3$ is uncovered as a result of the motion of nodal point, $p_1$ from the previous frame, $k$ to position, $p_1'$ in the predicted frame, $k+1$.

are not adequately modeled and updating the mesh to reflect the changes as follows. The motion vectors of the nodal points are first used to warp the mesh and its contents in the previous frame to obtain the current or predicted frame. The error between the warped or reconstructed frame and the actual current frame is subsequently used to identify the model-failure (MF) regions (i.e. by appropriate thresholding). The MF regions can be encoded and sent as additional overhead information while the mesh in the current frame is refined to adapt to the new image contents. This can be carried out by applying our segmentation scheme within the MF regions in order to update the mesh. Some of the issues related to the occlusion problem within the context of object-based coding are currently being explored [117].

5.4.5 Image Warping Prediction

The mesh tracking procedure just described allows us to determine the new positioning of the nodal points from frame to frame. In particular, the nodal points in the previous frame, $p_i$, are displaced by their motion vectors in order to estimate their new position, $p_i'$ in the predicted frame. Knowing the nodal point displacements also defines the correspondence between the triangles formed in the previous and predicted frames. Thus, in order to form the predicted frame, we must also transform or warp
the pixels within each triangle to their appropriate position. An affine transformation is used to determine each triangle-to-triangle mapping. Let $p_i = (p_{i1}, p_{i2})$ represent the pixel coordinates of the nodal points in the previous frame, and $p_i' = (p_{i1}', p_{i2}')$ the corresponding coordinates in the predicted frame. If $p_i$, for $i = 1, 2, 3$ are the coordinates of the three vertices of a triangle in the previous frame, and $p_i'$ their matches in the predicted frame, then the mapping function is defined as follows [104]

$$
\begin{bmatrix}
    p_{11}' & p_{12}' & 1 \\
    p_{21}' & p_{22}' & 1 \\
    p_{31}' & p_{32}' & 1
\end{bmatrix} =
\begin{bmatrix}
    p_{11} & p_{12} & 1 \\
    p_{21} & p_{22} & 1 \\
    p_{31} & p_{32} & 1
\end{bmatrix} \begin{bmatrix}
    a_{11} & a_{12} & 0 \\
    a_{21} & a_{22} & 0 \\
    a_{31} & a_{32} & 1
\end{bmatrix}
$$

(5.17)

The 6 transform coefficients, $a_{ij}$ for $i, j = 1, 2, 3$ are found by the expression

$$
\begin{bmatrix}
    a_{11} & a_{12} & 0 \\
    a_{21} & a_{22} & 0 \\
    a_{31} & a_{32} & 1
\end{bmatrix} = \frac{1}{\det(U)} \cdot \begin{bmatrix}
    p_{11}' & p_{12}' & 1 \\
    p_{21}' & p_{22}' & 1 \\
    p_{31}' & p_{32}' & 1
\end{bmatrix}
$$

(5.18)

where

$$
\begin{bmatrix}
    (p_{22} - p_{32}) & (p_{31} - p_{21}) & (p_{21}p_{32} - p_{22}p_{31}) \\
    (p_{32} - p_{12}) & (p_{11} - p_{31}) & (p_{12}p_{31} - p_{11}p_{32}) \\
    (p_{12} - p_{22}) & (p_{21} - p_{11}) & (p_{11}p_{22} - p_{12}p_{21})
\end{bmatrix}
$$

(5.19)

and

$$
\det(U) = p_{11}(p_{22} - p_{32}) - p_{12}(p_{21} - p_{31}) + (p_{21}p_{32} - p_{31}p_{22})
$$

(5.20)

The pixels, $n_i = (n_{i1}, n_{i2})$ within a particular triangle are subsequently warped to their new positions in the predicted frame using the corresponding transform coefficient values, $a_{ij}$ (i.e. determined from the nodal point correspondences above) as follows

$$
\begin{bmatrix}
    n_{i1}' & n_{i2}' & 1
\end{bmatrix} =
\begin{bmatrix}
    n_{i1} & n_{i2} & 1
\end{bmatrix} \begin{bmatrix}
    a_{11} & a_{12} & 0 \\
    a_{21} & a_{22} & 0 \\
    a_{31} & a_{32} & 1
\end{bmatrix}
$$

(5.21)
Bilinear interpolation is used to obtain the pixel values when the resulting coordinates are non-integer values. The pixel values within each triangle are successively transformed until all of the triangles have been warped. The facial region is then overlaid over the transformed image to form the final predicted frame. We also note, that in the case that a degenerate triangle is encountered in the predicted frame (i.e. due to the nodal point matching process), then the position of the nodal point causing the problem is altered so that the triangle connectivity is maintained. Its new position is determined by the bilinear interpolation of its two neighboring boundary nodal points.

The general structure of a predictive encoder/decoder scheme is shown in Figure 5.13. In conventional motion compensation, the predictor consists of the translational motion model which utilizes the standard block matching approach. The encoder codes the difference between the incoming frame and its prediction (i.e. prediction error) and transmits this information along with the motion vectors to the decoder. It is assumed that the predictor includes sufficient frame memory to store the previously reconstructed image. The decoder then reconstructs the predicted image using the received data. Our proposed warping prediction technique can easily be incorporated within this predictive codec by modifying the predictor as follows. At the encoder, the predictor block consists of the following: (i) the face localization component, (ii) the color segmentation scheme, (iii) the content-based mesh design step (i.e. nodal point selection), and (iv) the mesh tracking algorithm (i.e. nodal point motion vector estimation). The nodal point displacement vectors, and the inter or intraframe encoding of the facial region are sent to the decoder as overhead information. The predictor at the decoder is comprised of: (i) the color segmentation algorithm, (ii) the mesh generation scheme, (iii) the geometrical transformation block, and (iv) a decoder for the received facial information. The segmentation and mesh generation schemes are identical at both, the encoder and decoder. We note that the segmentation procedure is not carried out on each frame that is to be predicted, but only within the MF regions or when the prediction does not adequately model the original sequence (i.e. scene changes, poor mesh tracking, etc.). Further work is necessary in
Figure 5.13: General structure of a predictive encoder/decoder in a video compression scheme.
order to investigate these issues within the overall video coding scheme.

5.5 Experimental Results

The performance of our proposed motion compensated prediction technique was evaluated with several MPEG 4 video test sequences. A comparison of the warping prediction scheme was made with the conventional block matching motion compensator and the results of each step are presented below.

5.5.1 Color Image Segmentation

The segmentation results of the Carphone sequence (QCIF format - 176 x 144) are displayed in Figure 5.14. In part (a) of the figure, Frame 80 of the original image is illustrated while in part (b), the successful localization of the facial region is shown as determined from the previous chapter. The next three images of the figure depict the segmentation results within each of the primary regions outlined earlier (i.e. chromatic, achromatic, and transitional regions). In part (c), the chromatic regions are shown which constitute the most significant proportion of the image contents (i.e. \( \approx 40\% \)). The extracted achromatic regions are subsequently shown in part (d) and make up about 25% of the scene contents. The last of the three regions is the transitional area (10%) and only one region of significant size is extracted, as shown in part (e). Various statistics can be generated from these results (i.e. size, color, shape, orientation, spatial relationships, etc.) and these can be effectively employed for indexing in content-based storage/retrieval applications. The facial area which comprises about 15% of the image area can also reveal significant information about the scene [118]. In facial image databases (i.e. databases of employees, models, video newscasts, etc.) it can indicate: (i) whether the image is a close-up or full body shot, (ii) the skin-type (i.e. color) of the facial region, (iii) the orientation, (iv) the attributes of other facial characteristics (i.e. hair color, eyes, etc.), and (v) information about the type of objects that surround the facial region [118]. The result in part (f) demonstrates the effectiveness of the outlined technique in extracting the set
of arbitrarily-shaped regions with a sufficient level of detail (i.e. the segmented areas are pseudo-colored). Small regions were removed if their perimeter was less than the predefined threshold of 30 pixels. Setting a smaller threshold generates many additional smaller regions which may result in an oversegmented image. At the same time however, this may capture some additional detail present within the scene (i.e. the grass within the car window which gets lost). The experimentally selected threshold provides a good compromise between adequate detail preservation and minimal side effects from oversegmentation. However at this point, further refinement is still necessary in order to eliminate small regions that may be part of larger neighboring ones (i.e. small regions within the jacket area). In part (g), the final segmentation results are shown after region merging, where adjacent areas with similar color characteristics are joined and the number of regions reduced. A Euclidean distance threshold of 35 was experimentally determined and found to be suitable for this region merging process. Thus, in this final figure, we see that some of the small regions are effectively eliminated and merged with larger bordering regions. The post-processing operations included a $5 \times 5$ binary median filter, and a circular morphological structuring element. In future work, the region merging stage will be assisted by motion information so that the extracted regions can more closely correspond to semantically meaningful objects.

In Figure 5.15, the results of the Claire sequence (CIF format - 360x288) are depicted. The original of Frame 100 is shown in part (a), and the successful localization of the facial region in part (b). Once again, the arbitrary set of regions are effectively extracted in this less detailed scene. The chromatic regions are comprised of the jacket areas and the background, while the achromatic regions consist of the collars of the jacket (i.e. black) and the exposed part of the white shirt. We note that the Claire sequence corresponds to the simpler videoconferencing scenario in Figure 5.5 where one person is positioned against a stationary background.
5.5.2 Mesh Design

Having segmented the image into a set of arbitrarily-shaped regions, then a suitable mesh structure must be constructed to facilitate the warping process. Figure 5.16 (a) illustrates the selection of nodal points (i.e. the black pixels) along the boundaries of each region for the Carphone sequence. The selection scheme is carried out according to the method outlined in Section 5.4.3. The points are placed at evenly spaced intervals (i.e. every 10 pixels) and provide a reasonable piece-wise approximation of the boundary shapes. In part (b) of the figure, the resulting wireframe or mesh construction is shown. Most of the regions are modeled according to the mesh design in Figure 5.6. The background area in the top left corner of the figure (i.e. pink region), however, is an example of a region that is split vertically into two regions and then triangulated since it corresponds to the case in Figure 5.8. Several other regions such as the hair area are partitioned and triangulated as in Figure 5.7. We note that the facial area is not triangulated for reasons cited earlier.

A similar nodal point selection scheme is utilized for the Claire sequence and the results are shown in Figure 5.17 (a). The subsequent triangulation of these points is shown in part (b) and carried out according to the mesh design scheme of Figure 5.5 (i.e. moving person against a uniform stationary background).

5.5.3 Mesh Tracking

The constructed meshes in the previous section were temporally tracked by estimating the displacement vectors of the nodal points using the modified block matching technique and the proposed block structure of Figure 5.10. In Figure 5.18, the tracking performance is illustrated for the Carphone sequence. In part (a), the nodal points of the constructed mesh from Frame 80 of the original sequence is shown, while parts (b), and (c) depict the tracking results of these nodal points (i.e. yellow pixels) for Frames 85 and 90, respectively. The coefficient values chosen for the penalty factor, \( k \), and the distance threshold, \( c \) in equation (5.14) were 3, and 10, respectively. The motion vectors were restricted to integer values of \( \pm 16 \) pixels, in both, the horizon-
tal and vertical directions. This constraint, as mentioned earlier, ensures that longer motion vectors are not erroneously selected. In the Carphone sequence, this eliminates ambiguous matches in the lower left striped background area. This sequence is particularly difficult since the scene is shot inside a moving car where there are rapid occlusions, uncovering of objects, and spatially periodic textures (i.e. striped region of the car). Nevertheless, the modified block matching process performs extremely well even though the scene undergoes extensive motion between Frames 80 and 90. In part (c), the point matching process is slightly off at points around the hair region. This is attributed to the large motion between Frames 80 and 90, and the constraint enforced by the penalty function in equations (5.14-5.16). However, the penalty function is necessary in order to avoid erratic motion vectors and erroneous results in spatially periodic areas as described earlier. We also note that the nodal points which are covered by the facial regions are handled in the manner described in Section 5.4.4. Figure 5.19 (a) illustrates the segmented facial area (i.e. black area) of Frame 90 overlaid onto Frame 80. The nodal points, indicated by the white pixels, are the ones that are covered by the face in the predicted frame. The point matching process for these points follows the procedure outlined in Section 5.4.4. The nodal points which are initially on the boundary of the facial area and do not get covered by the face are constrained to lie on the boundary of the facial region in the predicted frame (i.e. the red outline shown in Figure 5.19 (b)).

The mesh tracking procedure is also shown in Figure 5.20 for the Claire sequence. In part (a), the nodal points from Frame 100 of the constructed mesh are shown, while in part (b), the tracking results are illustrated for Frame 110 of the sequence. Once again, the tracking performance is quite robust for this simpler type sequence.

5.5.4 Image Warping Prediction

The warping prediction scheme outlined in Section 5.4.5 was finally carried out on both, the Carphone, and the Claire sequences. The performance of our warping prediction approach was compared to the conventional block-based motion compensation technique (BMA). The full search algorithm using the minimum mean square error
criterion of the luminance component (i.e. Y part of the YUV space) was used as the block matching error criterion in the BMA method. The motion vectors in both schemes were restricted to have integer values in the range ±16, both, horizontally and vertically.

In Figure 5.21, the results for the prediction of Frame 85 (i.e. determined from the original Frame 80) of the Carphone sequence is shown. In parts (a), and (b), the original images of Frame 80, and 85, are shown, respectively while in part (c), the results from the standard block matching technique (BMA) are shown. The blocking artifacts in the conventional case are very pronounced, particularly in the facial region, resulting in an unacceptable picture quality. The results of the warping prediction scheme in part (d) on the other hand, clearly demonstrate the improved performance. The image is free from visually degrading artifacts. The facial area shown in the figure is coded in a lossless intraframe manner. Table 5.2 below, summarizes the advantages and disadvantages of each technique for several different categories. From

Table 5.2: Summary of Motion Compensated Prediction for Frame 85 of the Carphone sequence.

<table>
<thead>
<tr>
<th>Attributes</th>
<th>Motion Compensation Technique</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Standard Block Matching</td>
<td>Image Warping Prediction</td>
</tr>
<tr>
<td>MSE</td>
<td>237.3</td>
<td>197.7</td>
</tr>
<tr>
<td>MAE</td>
<td>11.5</td>
<td>10.2</td>
</tr>
<tr>
<td>Overhead</td>
<td>99 Motion Vectors +</td>
<td>158 Motion Vectors +</td>
</tr>
<tr>
<td></td>
<td>60948 bytes for Prediction Error</td>
<td>9297 bytes for Facial Region</td>
</tr>
<tr>
<td></td>
<td>Total Overhead = 61146 bytes</td>
<td>Total Overhead = 9613 bytes</td>
</tr>
<tr>
<td>Complexity</td>
<td>Encoder</td>
<td></td>
</tr>
<tr>
<td></td>
<td>1) Motion Estimation</td>
<td>1) Color Segmentation</td>
</tr>
<tr>
<td></td>
<td>Decoder</td>
<td>2) Mesh Generation</td>
</tr>
<tr>
<td></td>
<td>1) Translational Model</td>
<td>3) Motion Estimation</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Encoder</td>
</tr>
<tr>
<td>Provision for</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Content-based Functionalities</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Visual Quality</td>
<td>Poor</td>
<td>Good</td>
</tr>
</tbody>
</table>
the table, we see that both, the Mean Square Error, (MSE) and the Mean Absolute Error, (MAE) values are lower for the warping prediction scheme, as would have been expected from the visual quality of Figure 5.21. The overhead required in the BMA case (i.e. transmission at the encoder) is 99 motion vectors (i.e. 198 bytes since 2 bytes are required for each motion vector) plus the prediction error since the quality of the motion compensated prediction alone is not acceptable. There are 20316 non-zero pixels in the prediction error that must also be sent (i.e. 3 bytes per color pixel). Thus, a total of 61146 bytes are required for overhead in the BMA case. In our warping prediction technique, 158 motion vectors are necessary (316 bytes), plus 9297 bytes required to intraframe encode the facial area. Thus, the total overhead is 9613 bytes which is a significant reduction from the conventional case. If there are only small changes in the facial region (i.e. between frames), then these can be interframe encoded to reduce the overhead. We have also found that the number of motion vectors can be further reduced by utilizing a better nodal point placement strategy based on the curvature of the region boundaries. Figure 5.22 indicates a placement scheme that reduces the necessary motion vectors from 158 to 128, and also provides a better representation of the boundary regions. Future work will include this approach into the video coding scheme.

Next, we look at the complexity of the two schemes. In the BMA method, motion estimation is required at the encoder while a translational prediction model is necessary at the decoder. In the warping prediction technique, the encoder consists of: (i) a color segmentation algorithm (face localization and general segmentation), (ii) a mesh generation scheme (i.e. nodal point placement and triangulation), and (iii) a nodal point motion estimation method. At the same time, the decoder is comprised of: (i) the same color segmentation and mesh generation algorithm, and (ii) an affine warping prediction model. Thus, there is a slight increase in the computational complexity of our proposed method over the conventional case. Nevertheless, the overall complexity of the proposed method is still reasonable due to the speed of the HSV segmentation approach and the low complexity of the affine transformations in the triangular mesh model. However, we note that the general segmentation process (i.e.
segmentation into arbitrarily-shaped regions) is not required for each frame that is to be predicted. It is only required for the initial frame, and the model failure areas (i.e. scene changes, and motion failures).

The second last row of Table 5.2 gives us an indication of additional features that may be provided with each approach. We see that the warping prediction method is suitable for content-based functionalities (i.e. storage/retrieval, editing/compositing, etc.) due to its object-based approach. The technique can provide information about the color content within the image, facial information, spatial relationships between objects, etc. The block-based method on the other hand, cannot provide this type of functionality. Finally, the last row of the table indicates that the visual quality of the warping prediction scheme is significantly better than the BMA method as was also demonstrated in Figure 5.21.

The motion compensation schemes above were also compared by determining the prediction of Frame 90 for the same Carphone sequence, once again determined from the original Frame 80 of the sequence. In this case, the two images are spaced 10 frames apart with greater motion between them. The prediction results are shown in Figure 5.23. The original images of Frames 80 and 90 are shown in parts (a), and (b), respectively. The output from the BMA method is shown in part (c). The results for the conventional case are severely degraded when greater motion is exhibited. Once again, the warping prediction method in part (d) yields results that are free from the annoying blocking artifacts. There are slight geometrical distortions which, nevertheless, are more acceptable to a human observer. Slight deformations can be observed in the hair area, as well as the jacket region. The head region undergoes significant motion between frames which is not adequately modeled due to the constraint on the distance of the motion vectors. As a result of this, the hair region is shifted and stretched. Table 5.3 below provides a summary of the results for this particular case. It is interesting to see that the MSE, and MAE values are both higher for the warping predictor, despite the superiority in the visual quality. This may be attributed to the shifting of the hair region in the warping method. This is a good illustration of the unsatisfactory performance of the numerical MSE, and MAE values as robust
Table 5.3: Summary of Motion Compensated Prediction for Frame 90 of the Carphone sequence.

<table>
<thead>
<tr>
<th>Attributes</th>
<th>Motion Compensation Technique</th>
<th>Standard Block Matching</th>
<th>Image Warping Prediction</th>
</tr>
</thead>
<tbody>
<tr>
<td>MSE</td>
<td></td>
<td>341.5</td>
<td>385.7</td>
</tr>
<tr>
<td>MAE</td>
<td></td>
<td>14.6</td>
<td>15.9</td>
</tr>
<tr>
<td>Overhead</td>
<td></td>
<td>99 Motion Vectors + 60402 bytes for Prediction Error</td>
<td>158 Motion Vectors + 8853 bytes for Facial Region</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Total Overhead = 60600 bytes</td>
<td>Total Overhead = 9169 bytes</td>
</tr>
<tr>
<td>Complexity</td>
<td>Encoder</td>
<td>1) Motion Estimation</td>
<td>Encoder</td>
</tr>
<tr>
<td></td>
<td>Decoder</td>
<td>1) Translational Model</td>
<td>Decoder</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>1) Color Segmentation</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>2) Mesh Generation</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>3) Motion Estimation</td>
</tr>
<tr>
<td>Provision for Content-based</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Functionalities</td>
<td></td>
<td></td>
<td>Yes</td>
</tr>
<tr>
<td>Visual Quality</td>
<td>Extremely Poor</td>
<td>Satisfactory</td>
<td></td>
</tr>
</tbody>
</table>

measures of visual quality. Thus, the need for better perceptual measures warrants further research into this area. The necessary overhead in this case is similar to the prediction of Frame 85. A slight increase in the number of motion vectors are required in the warping predictor but the quality is sufficient so that the prediction error is not necessary. Only 8853 bytes are necessary to intraframe encode the facial area (i.e. shown coded losslessly in the figure). The BMA method yields extremely poor prediction results and, thus requires the addition of the prediction error. The remaining attributes are the same as in the case for the prediction of Frame 85. Thus, we see that the warping predictor performs satisfactorily even in the case of greater and more complex motion as in this latter example.

Finally, we apply the warping prediction scheme to observe its performance on the simpler Claire CIF size (360 × 288) video sequence. The prediction of Frame 110 is made from the original of Frame 100. In parts (a), and (b) of Figure 5.24, the original
images of Frame 100, and 110 are shown, respectively. Part (c) of the figure illustrates the prediction results of the BMA method. Once again, the reconstructed or predicted image is severely degraded in the conventional case. The block-based approach fails to adequately model the motion between frames. In part (d), the output from the warping prediction method is shown, clearly demonstrating the improvement in the visual quality of the results. Only a few minor degradations can be observed such as the left collar area, and the hair region to the right of the facial area. The facial region is once again shown losslessly intraframe encoded. Table 5.4 summarizes the results for this sequence. The numerical MSE and MAE measures are both considerably lower for the warping approach over the standard BMA method. Furthermore, only 240 motion vectors are required in the proposed technique versus 320 for the conventional one. This is a significant reduction in modeling the motion field. Additional overhead requirements for the warping predictor include approximately 13800 bytes for the facial area, whereas the BMA method demands over 219 KB for the prediction error. In the end, the visual quality of the proposed method is significantly better than the BMA scheme as also seen from Figure 5.24. Thus, overall, the warping predictor outperforms the BMA approach considerably in all categories. The improvements are even greater for the case of the simpler videophone-type sequence where a single person is moving against a stationary uniform background as in the latter example.

<table>
<thead>
<tr>
<th>Attributes</th>
<th>Motion Compensation Technique</th>
<th>Standard Block Matching</th>
<th>Image Warping Prediction</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>200.0</td>
<td>129.4</td>
</tr>
<tr>
<td>MSE</td>
<td></td>
<td>8.0</td>
<td>7.1</td>
</tr>
<tr>
<td>MAE</td>
<td></td>
<td>320 Motion Vectors +</td>
<td>240 Motion Vectors +</td>
</tr>
<tr>
<td></td>
<td></td>
<td>219201 bytes for Prediction Error</td>
<td>13824 bytes for Facial Region</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Total Overhead = 219841 bytes</td>
<td>Total Overhead = 14304 bytes</td>
</tr>
<tr>
<td>Visual Quality</td>
<td>Extremely Poor</td>
<td>Good</td>
<td></td>
</tr>
</tbody>
</table>

Table 5.4: Summary of Motion Compensated Prediction for Frame 110 of the Claire sequence.
5.6 Conclusions

A novel image warping technique using spatial transformations was introduced for motion compensated prediction of videophone-type sequences. The method we have proposed can also be applied to general video scenes and is suitable for future object-based video coding environments as in the MPEG 4 and MPEG 7 standardization efforts. The approach we have presented consisted of the following components: (i) face localization, (ii) color segmentation, (iii) mesh generation, (iv) mesh tracking, and (v) image warping prediction.

A fast color segmentation scheme using the perceptual HSV color model was used to effectively partition a videophone-type sequence into a facial area and a set of arbitrarily-shaped regions. Face localization was outlined in the previous chapter, while general segmentation using a recursive 1D histogramming procedure was addressed in this chapter. The technique was found to be robust, computationally inexpensive, and it also allowed the patch or region boundaries to match the scene content, enabling a more accurate motion estimation and compensation. Next, a wireframe mesh structure was devised by using a suitable nodal point placement and triangulation scheme. An efficient, triangular mesh model was generated so that the number of degenerate triangles were minimized during the tracking process. The effect of this process was to place more nodal points in the detailed areas of the image, and fewer points in uniform regions. A modified block matching technique was subsequently utilized to obtain the nodal point motion vectors, and effectively track the mesh structure.

Steps (i) to (iv) were carried out at the encoder (general segmentation was only carried out initially, and in (MF) motion failure areas), and the motion vectors along with the encoded facial area were sent as overhead to the decoder. The facial region was separately encoded to avoid distortions of the facial features which contribute significantly to the intelligibility of the signal. At the decoder, the same segmentation (only required initially and in MF areas), and nodal point selection/triangulation algorithm were used to obtain the mesh structure. The nodal points were spatially
shifted using the motion vectors, and the predicted image was formed with affine warping transformation. A significant subjective improvement (free of blocking artifacts) was found in the warping predictor when compared to the conventional block-matching approach. The results were even more pronounced when trying to predict scenes that exhibited greater motion. The high visual quality was achieved with a similar transmission overhead, and only a moderate increase in the computational complexity of the coding scheme. Experimental results were obtained for complex videophone sequences with detailed moving backgrounds, as well as simpler head-and-shoulders scenes against stationary, uniform backgrounds. In addition to the robust performance, the suggested technique can also be used to provide information about the facial region as well as the proportion of chromatic and achromatic content within the scene for future content-based functionalities. The effectiveness of the proposed warping prediction scheme, and its potential for a more suitable content-based representation is encouraging for future object-based video coding environments.
Figure 5.14: Segmentation of the Carphone sequence: (a) Original Frame 80, (b) extraction of the facial region, (c) the chromatic regions, (d) the achromatic regions, (e) the transitional chromatic/achromatic regions, (f) segmentation before region merging, and (g) final segmentation after region merging.
Figure 5.15: Segmentation of the Claire sequence: a) Original Frame 100 of the Claire sequence, b) Extraction of the facial region, c) Final Segmentation after Region Merging.
Figure 5.16: (a) Nodal point selection along the boundaries of the segmented regions in Frame 80 of the Carphone sequence, and (b) mesh generation.

Figure 5.17: (a) Nodal point selection along the boundaries of the segmented regions in Frame 100 of the Claire sequence, and (b) mesh generation.
Figure 5.18: Nodal point selection along the boundaries of the segmented regions in Frame 80 of the Carphone sequence, (b) tracking of nodal points in Frame 85, and (c) the tracking of nodal points in Frame 90.

Figure 5.19: (a) Nodal points (i.e. white pixels) covered by the facial area (i.e. black area) of the predicted frame, and, (b) pixels on the boundary of the facial area in the previous frame which are not covered by the face in the predicted frame are constrained to lie on the boundary in the predicted frame (i.e. the red outline).
Figure 5.20: (a) Nodal point selection along the boundaries of the segmented regions in Frame 100 of the Claire sequence, and (b) tracking of nodal points in Frame 110.
Figure 5.21: Motion compensated prediction of the Carphone sequence: (a) original Frame 80, (b) original Frame 85, (c) motion compensated prediction of Frame 85 using standard block matching (BMA), and (d) motion compensated prediction of Frame 85 using warping prediction.

Figure 5.22: An efficient nodal placement strategy that reduces the number of necessary motion vectors.
Figure 5.23: Motion compensated prediction of the *Carphone* sequence: (a) original Frame 80, (b) original Frame 90, (c) motion compensated prediction of Frame 90 using standard block matching (BMA), and (d) motion compensated prediction of Frame 90 using warping prediction.
Figure 5.24: Motion compensated prediction of the Claire sequence: (a) original Frame 100, (b) original Frame 110, (c) motion compensated prediction of Frame 110 using standard block matching (BMA), and (d) motion compensated prediction of Frame 110 using warping prediction.
Chapter 6

Conclusions and Future Directions

In this thesis, spatial and temporal prediction techniques were investigated for a newer generation of video compression schemes which are object-based. The focus in object-based video coding is to represent the image by its structural features such as contours and regions as opposed to the conventional block-based methods. This approach, not only provides for a more efficient coded representation, but it can also support future content-based functionalities that will allow for better ways to interact and exchange visual information. The tremendous interest amongst groups in both, industry and academia, have aimed to meet these challenges through their joint efforts in establishing the future MPEG 4 and 7 video coding standards.

The thesis work can be essentially divided into two components: (i) spatial image interpolation techniques, and (ii) segmentation-based motion compensated prediction. The first two chapters investigate the use of structural as well as statistical models for interpolating digital images that have been spatially down-converted for compression purposes or simply for improved-definition image zooming. This is useful for editing/compositing in object-based video and also in frame-based compression schemes where storage and transmission of video is allowed in an uncompressed form. The latter two chapters of the thesis examine the motion compensated prediction problem within the framework of an object-based coder. The focus is predominantly on videophone-type sequences which consist of people (i.e. head and shoulders) moving against complex backgrounds, however, the approach can be generalized for arbitrary
scenes. One chapter examines the aspect of facial image segmentation while the final one investigates an arbitrary segmentation and motion compensated prediction scheme.

In chapter 2, the spatial interpolation of color images was carried out using nonlinear filter structures based on vector order statistics (OS). Vector FIR-median hybrid (VFMH) filters were chosen from this OS class of filters due to their robustness, preservation of edge information and image details, and their ability to exploit the existing correlation between colors. Several VFMH filter designs were implemented and compared to the conventional linear schemes. The problems associated with the linear techniques are well known, that of blockiness, aliasing, and blurring of the interpolated image. The experimental results indicated that the VFMH filters performed better, both, quantitatively and qualitatively than the linear methods, and also when the OS filters were applied to each color channel independently. The VFMH filters reduced the artifacts found in the linear approaches, preserved the step edges and image details better than their counterparts, and retained the color correlations due to their vector approach. Operating vectorially also offers the advantage (over independent channel processing) of reducing impulsive-type noise which may be the case if video scaling is carried out in real-time from a broadcast transmission. The computational complexity of these nonlinear filters is slightly increased over their linear counterparts, however, they are still feasible for implementation, particularly in hardware.

In chapter 3, statistical techniques using Gibbs random field models were used to obtain an interpolated image from a downsampled version. Three iterative techniques, that of the Gibbs Sampler, Simulated Annealing, and the Iterative Conditional Mode were implemented based on a set of estimated model parameters. Parameters were determined over small rectangular blocks within the image. Even though a simplified model with a small number of parameters was selected, the computational complexity was high which demanded offline processing. However, at run-time (i.e. having determined the parameters) the iterative techniques were approximated by a non-iterative 3-pass filter that allowed real-time performance. The experimental results indicated
that the statistical methods did not exhibit the usual artifacts present in the linear approaches, but did incur some loss in image detail and some noise-like artifacts. Nevertheless, the results were quite reasonable and promising. In future work, the parameter estimation process should be applied over segmented regions rather than square blocks. This would avoid trying to fit one set of parameters to a block which may have different statistics within it. An additional improvement may be realized if the size of the neighborhood is increased and a larger number of clique types included. This is now more feasible with the rapid and continuous improvements in computational power.

The second component of the thesis focused on the temporal prediction of video sequences based on a partition of the scene into arbitrarily-shaped regions rather than the conventional square blocks. Image sequences consisting of head-and-shoulders people against complex backgrounds were of primary interest. Thus, in chapter 4, the automatic location and tracking of the facial region was investigated for these videophone-type sequences. Extraction of the facial area not only allows a video coder to place more emphasis on the facial region but it can also provide content-based storage and retrieval functionalities in future coding environments. The attributes of color and shape were utilized in devising a three-stage segmentation scheme for the extraction of the facial region. The initial segmentation was provided by a color processing module that operated in the HSV color space. A number of fuzzy membership functions were subsequently used to select the facial area in the case of falsely included regions. The proposed technique was found to be robust with regards to facial shape, size, skin-type, motion, and lighting conditions. Furthermore, the computational complexity of the scheme was low due to the one-dimensional processing of the HSV histograms, and the binary nature of the post-processing operations performed. Facial image segmentation is an area of interest to many research groups for various application purposes, including, video coding, face identification, and image and video retrieval, to name a few. An extension of the work carried out here could focus on a number of different aspects. The fuzzy membership functions utilized here were primarily tuned for videophone-type applications. Future work could focus on
arbitrary scenes which contain multiple faces in arbitrary locations and sizes. One could investigate the performance of the proposed method on these more general type of scenes and accordingly adjust or devise new membership functions and thresholds. Membership functions based on facial features may provide useful information for this purpose. Facial extraction is also of paramount interest from the retrieval point of view. Indexing and retrieval of facial images based on a number of features such as skin/hair/eye color, facial size and orientation, and spatial relationships is a new and untackled area. We have determined some very promising preliminary findings [118]. Finally, the effects of lighting, and occlusions (i.e. objects, beards, etc.) on the segmentation process must be studied further in greater detail.

In chapter 5, a novel region-based image warping technique using spatial transformations was proposed for motion compensated prediction of videophone-type sequences. A fast color segmentation scheme was introduced to effectively partition an image into a set of arbitrarily-shaped regions plus the facial area obtained from the previous chapter. A triangular mesh model was subsequently constructed for the segmented image using a proposed nodal point placement and triangulation scheme. Modified block matching was then used to determine the nodal point motion vectors, and effectively track the mesh structure. Motion compensated prediction was finally achieved by utilizing the motion vectors and an affine warping transformation. A significant subjective improvement was found in the warping predictor when compared to the conventional block-matching approach. Errors in the proposed method resulted in geometrical distortions which are less disturbing to the eye than the blocking artifacts that typically form in the standard block-matching techniques. The computational complexity of our object-based approach is moderately increased due to the segmentation and facial extraction process. However, the segmentation procedure is only carried out for one frame and then the mesh is subsequently tracked from frame to frame (i.e. with only periodic updates in the motion failure areas). The number of transmitted motion vectors for both, the warping predictor and the standard block-matching method are on the same order. However, the unacceptable quality in standard block-matching (i.e. especially when the scene undergoes extensive and
complex motion) requires that the prediction error must also be sent as additional overhead which amounts to a significant number of bits. The warping predictor, on the other hand, requires that the facial area information be transmitted (i.e. for videophone-type applications) which is typically a smaller number. The proposed scheme is also advantageous as it can be used to provide information about the facial region as well as the content within the scene useful for searching and retrieval functions. The results of our proposed method are very encouraging demanding further research to exploit its full potential. The object-based approach must be incorporated into a complete video coding framework (with lengthy sequences and different types) in order to make concrete comparisons of compression efficiency, picture quality, and computational complexity. The issue of occlusion (i.e. covered and uncovered objects) must also be explored in greater depth. We have taken this into account for the facial region but not in the general case. Further work is also recommended in investigating a joint motion/color segmentation scheme that may provide a better representation of the motion boundaries and also allow for higher level semantics to be determined. It may also be worth examining a texture/color segmentation scheme for highly textured scenes where color alone may result in an oversegmentation of the image. Finally, the derivation of content-based descriptions for indexing and retrieval purposes is another new and vast area of research that is definitely worth exploring.
Bibliography


