

13

People Objects: 3-D Modeling of Heads in Real-Time

by
Thomas E. Slowe

B.S., Electrical Engineering
Rutgers University, College of Engineering
June 1996

Submitted to the Program in Media Arts and Sciences,
School of Architecture and Planning,
in partial fulfillment of the requirements for the degree of
MASTER OF SCIENCE IN MEDIA ARTS AND SCIENCES
at the
Massachusetts Institute of Technology
September 1998

© Massachusetts Institute of Technology, 1998
All Rights Reserved

Handwritten marks: a vertical line, a small 'r', and two 'A's.

Signature of Author:

Program in Media Arts and Sciences
7 August 1998

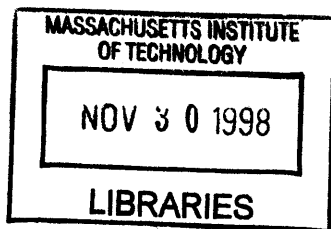
Certified by:

V. Michael Bove, Jr.
Principal Research Scientist
Program in Media Arts and Sciences
Thesis Supervisor

Handwritten signature of V. Michael Bove, Jr.

Accepted by:

Stephen A. Benton
Chairperson
Departmental Committee on Graduate Students
Program in Media Arts and Sciences



Handwritten mark: "etc"

People Objects: 3-D Modeling of Heads in Real-Time

by
Thomas E. Slowe

Submitted to the Program in Media Arts and Sciences,
School of Architecture and Planning,
on August 7, 1998
in partial fulfillment of the requirements for the degree of
Master of Science in Media Arts and Sciences

Abstract

In this thesis project, a real-time face tracking system and a real-time head rendering system will be developed to be used in conjunction. Three cameras will be employed, although the system is both upwardly and downwardly scalable. The face finder contains a rotation invariant, gross search feature finder, a face validation routine, and a real time tracker that uses local area searches given past feature estimates. The face renderer operates using a generic three dimensional head model. The system is optimized for use in virtual three dimensional interaction.

Thesis Supervisor: V. Michael Bove, Jr.
Title: Principal Research Scientist

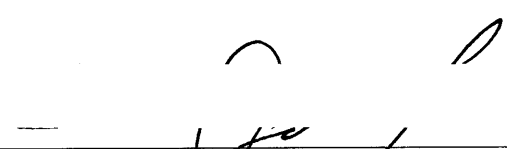
This work was supported by the Digital Life consortium.

People Objects: 3-D Modeling of Heads in Real-Time

by
Thomas E. Slowe

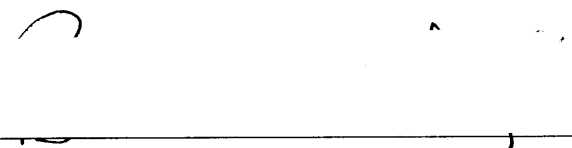
The following people served as readers for this thesis:

Reader:



Aaron Bobick
Assistant Professor of Computational Vision
Program in Media Arts and Sciences

Reader:



Bruce Blumberg
Assistant Professor of Media Technology
Program in Media Arts and Sciences

Acknowledgments

I deeply appreciate the help of all that have influenced me.

Mike Bove for providing opportunity to work at the Media Lab on some wonderfully interesting problems. I am thankful for the friendly tugs back to earth.

Aaron Bobick for offering challenges in both computation and perception.

Bruce Blumberg for forcing me to justify upon every front, my thesis work as a whole.

Henry Holtzman for helping to provide me with some wonderful computers on which to do research.

Stefan Agamanolis for helping to answer many questions regarding the use of Isis, general programming, and the difference between a hyphen and a subtraction sign.

Bill Butera for helping me to optimize code, good advice in areas both technical and practical, and for keeping my spirits up.

Yuri Ivanov for always pointing me in the right direction.

Alex Westner for having a beer with me after many a long day.

Emily Katherine Slowe, my sister, for whacking me in the head when I confuse the obvious with the obscure.

James Elwood and Betty Jean Slowe, my parents, for teaching me how to think with an open mind.

Contents

1	Introduction	7
1.1	Defining the Task	8
1.1.1	Quantification of the Problem	8
1.1.2	Applications	9
1.2	Past Approaches	14
1.2.1	Face Detection and Tracking Systems	14
1.2.2	Face Rendering Systems	21
1.3	System Overview	25
1.3.1	Overview of Face Finding	26
1.3.2	Overview of Face Rendering	27
2	Face Finding	29
2.1	System Description	30
2.1.1	Face Detection	32
2.1.2	Estimating Head Orientation	35
2.1.3	Face Validation	37
2.1.4	Tracking with Warped Templates	39
2.2	Learning a Face	40
3	Face Rendering	42
3.1	System Description	42
3.1.1	Warp Details	44
3.1.2	Confidence Maps	46
3.1.3	Online Adjustment of Maps	49
4	Implementation Specifics and Future Work	50
4.1	Hardware	50
4.2	Software	51
4.3	Real Time Considerations	51
4.4	Future Work	52

List of Figures

1.1	Physical Hardware for Teleconferencing	11
1.2	Viewpoint Problem in Telecommunication	12
1.3	Overall System Diagram	25
1.4	Representative Physical Setup	26
2.1	Face Finding System Diagram	30
2.2	Decision Making of the Manager	31
2.3	Feature Estimates Over Several Frames	34
2.4	The Head Model	36
2.5	A Representative Warped Face	38
3.1	Face Rendering System Diagram	43
3.2	The Model Unwrapped	45
3.3	A Representative Confidence Map	47
3.4	The Flow of Composition	48

Chapter 1

Introduction

Presented is a scheme for producing three dimensional people objects in real time. Human head detection, tracking, rendering, and compositing algorithms that have been implemented in software on conventional hardware, will be described. Systems of this type are popular subjects in current research literature. Many blocks of the system are open problems in computer vision. Needless to say, so is the thesis topic.

The creation of people objects in real-time requires knowing where each person is and what they look like from every perspective. Faces can be found and tracked using a face finder. Faces can be rendered if one maintains prior knowledge of the structure of a human head. The function performed by each block of the system must maintain real time performance during normal operation.

The justification for this thesis work will be shown to be driven both by application and novel approaches to the system design. In the next section, a review of the objective of this thesis project will be given. In addition, it will be clearly demonstrated that there is an overwhelming number of valuable applications for this technology. In the section following, past work in this area and areas similar, will be discussed. In the last section, a review of the approach in this thesis will be given.

1.1 Defining the Task

In this section, the problem, mode of solution, and area of research will be justified as academically and pragmatically worthy.

This technology has a wide variety of applications, that will be reviewed in the next subsection. It is generally held that, each application requires a specially designed face finding and modeling system. This is a safe assumption, given the use of conventional mass-produced hardware, because real time constraints are too rigid to allow for general purpose systems. Naturally, the growth in processor speed or the incorporation of specialized hardware, such as that made by Cognex [cogn98] would relax or eliminate this consideration.

In turn, processor speed could not eliminate research in this area, because the core of the problem is the quantification of the human appearance. We find it very difficult to concisely describe what the important distinctions are between the appearance of a face and that of other objects. It is even more difficult to describe the important distinctions between different faces.

1.1.1 Quantification of the Problem

It is necessary to describe what functions the system will and won't perform.

The system must render virtual viewpoints of human faces in real time, given live color video that contains faces at arbitrary orientations. This requires the system to precisely identify facial features and maintain estimates of their locations in real time. In addition, the system must alter the texture maps of the face such that the illusion of a true viewpoint is produced at arbitrary orientation. The system must alter viewpoint, it does not need to remember what the individual looks like.

The entire system should operate in real time and be hands free, whereby once the cameras

and appropriate lighting are set up, and the algorithm is initiated, it should run without input from the user or a human controller. The system should be seamless, whereby users of the system should not need to know of the system's existence.

The system must understand if there is a face within view or not. It must find human facial features and then track them in real time, where features are defined as eyes, ears, the nose and mouth. If there is not a face present, it must continue to look until one appears. The face should not have to be oriented in any special way, with reference to the cameras, at any time. If the individual moves out of view of all cameras the system must find them again if they come back into view. The system cannot be expected to maintain real time operation if there are drastic changes in the true feature location between two consecutive frames of video. In addition, the system can assume that if a face comes within view the person will not move much until the system has been initialized. Initialization should only take a few seconds.

The system should operate with as few as one camera and scale to use many more. The cameras should not have to be arranged in any special way, although the system should render with greater resolution and support a wider range of output orientations if each camera has a unique view of the face.

1.1.2 Applications

Day to day, it is critical that we track, recognize and even model each others faces in real time. In fact, some occupations (such as positions in security, the military, and sales) are almost solely based upon these functions. It follows that there is a multitude of useful applications for technology that can perform some of these functions, even in a limited manner.

As will be described in the next section many of the applications described herein have been realized by use of physical apparatus to track features. Some systems, such as those used

for virtual interaction, can be greatly enhanced by a hands free(no physical apparatus) face tracking and modeling system. Systems that perform face recognition, expression determination, and three dimensional audio applications, require real time updates of head location and the lack of invasiveness of a hands free face tracking system would enhance the value of these applications.

Two Dimensional Interaction

The physical setup of a typical virtual interaction system can be found in Figure 1.1. A camera is present to film each participant for display on the monitor. Each person speaks into a microphone, while listening to the others with a headset.

Teleconferencing is one form of virtual interaction, where the live video feed with a particular user in view is encoded and compressed to the point where the information can be transmitted close to real time, given the bandwidth requirements. A teleconferencing system is different from a normal telephone system in that you can see the person you are talking to in addition to the fact that multiple users at each site are allowed.

Compression of the video signals could be increased by rendering the participants at the receiving side of transmission instead of having to send a texture map of them every thirtieth of a second. This would require the measurement of the orientation of the head and changes in facial expression. A system targeted at this application can be seen in [mcco97].

Telepresence applications, such as Reflection of Presence[agam97a], represent other forms of virtual interaction. The users of the system exist in a modifiable two dimensional virtual space. The system is collaborative and the environment it provides is immersive and common among participants[agam97a]. These systems differ vastly with teleconferencing systems in that the video and audio channels received from each site are manipulated to aid social interaction.

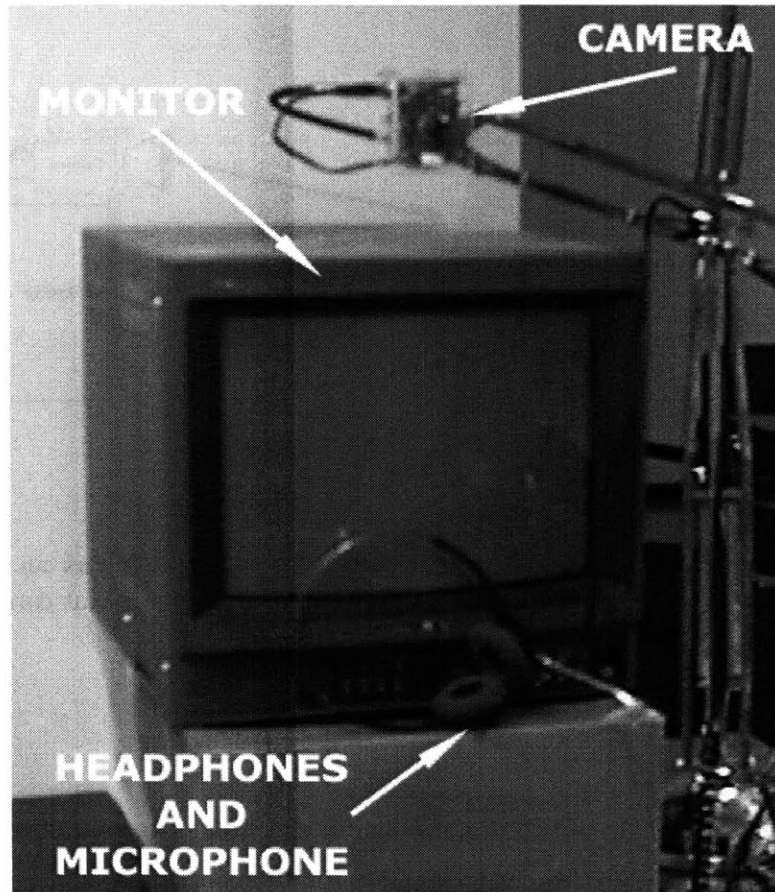


Figure 1.1: The camera is placed above the display monitor to capture the participant while they interact. The headset microphone must be worn by each participant to record the audio and provide the user with speech from the other participants.

Both telepresence and teleconference systems are subject to a common problem, viewpoint errors. As can be seen in Figure 1.2, the participant is looking at the monitor where the line of sight of the participant is several degrees off from the optical axis of the camera. Each view of the individual can be rendered to align the two axis. The face will not always be rendered looking into the camera, but simply adjusted so that when the participant is looking into the monitor it will appear that they are looking into the camera.

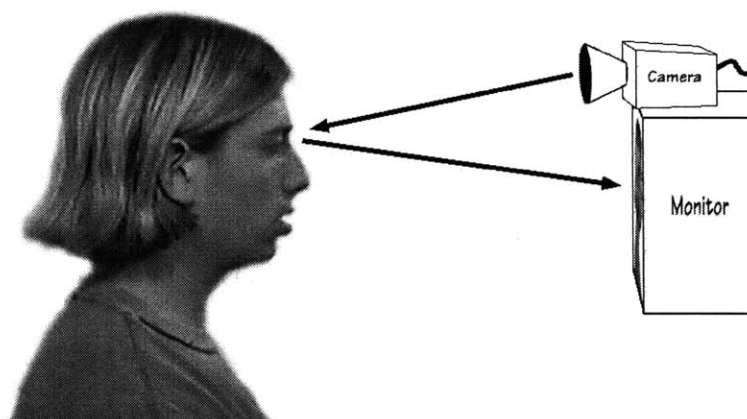


Figure 1.2: It can be seen that the participant's line of sight incurs an angle of several degrees with the optical axis. This can be corrected using a viewpoint correction scheme.

Three Dimensional Interaction

Despite these enhancements of two dimensional interaction systems, full social interaction, as it takes place in real space, cannot be simulated. Without the simulation of movement in three dimensional space it becomes difficult for people to interact in a comfortable manner. There is no natural way in which people can conduct side conversations. The belief is that a system that incorporated some type of three dimensional modeling would greatly enhance human to human remote interaction.

A constraint imposed by three dimensional interaction is that viewpoints from every vantage point of the scene and the people within it must be provided. A large body of work exists in the area of three dimensional modeling of scenes. Much of this work has been to provide

for real-time rendering of virtual environments. In [beck95], models of three dimensional spaces were semi-automatically extracted from several uncalibrated still images. Much of the work in this area assumes that “scenes that contain every-day man-made objects often possess sets of parallel lines and orthogonal planes”[beck95]. This is a powerful constraint in three dimensional modeling.

The fabrication of people objects provides the other half of the technology to enable seamless virtual three dimensional interaction. People objects can provide the real time virtual viewpoints of the participants while the scene is rendered using proven methods.

Face Recognition and Expression Determination

Face recognition as its name implies is the study of systems that attempt to distinguish between different human faces. Expression determination also is a recognition problem, where the goal is not to determine the differences between human faces, but the similarity in facial deformities that represent face expressions.

Face recognition and expression determination systems require accurate estimates of the facial features. Systems such as these are vastly more versatile if they can operate in real time which necessitates real time updates of feature locations. These applications then become practical to embed in products we use everyday such as ATM cash machines and cars.

Three Dimensional Audio

Three dimensional audio systems produce sounds at arbitrary locations based upon the location of the ears of the user[gard97]. The purpose of this is to produce an immersive interactive sound environment in virtual space as in real space. As implied above, real time estimates of the location of the ears is required.

1.2 Past Approaches

There have been numerous works in the research area of three dimensional face tracking and modeling in recent years. Because each system is geared towards a particular application it is difficult to compare relative performance of different systems. Below is found a review of the works that influenced the design of the system presented in this work.

1.2.1 Face Detection and Tracking Systems

Face detection and tracking systems are different only in that tracking systems operate in real time. Typically, real time performance is achieved by sacrificing certain resources such as accuracy, allowance of drift, variance to lighting, scale, in addition to many others. Nonetheless, the two system types must be presented in conjunction due to vastly similar approaches. I will note the computational complexity of each algorithm when relevant to the discussion.

In [lam98], it was noted that face analysis systems tend to fall in one of two groups, either analytic or holistic [lam98]. Nonetheless, I feel there are two additional groups; color systems and systems that require physical apparatus. A review of each group will be given.

Analytic Systems

Analytic approaches attempt to quantify the face by describing mathematically, physical parts of recognizable features such as eyes and nose. There is a natural progression from first to last paper described in this subsection. The first few papers found the parts of each feature quite well if the search was started near the the feature. This will not always be the case, especially when the system is planted in a practical application. Thus global constraints were employed, and system performance became drastically more robust.

In [yuil92], a very detailed deformable template(or model) of the eye and mouth were employed. Two templates were constructed for the mouth; one for a closed mouth, one for an open mouth. Descriptions were in terms of what people describe as parts of the human features. In other words, listed parts of the eye included regions such as the iris, pupil and whites. Certain edge detectors were used to filter the input. Energy functions, dependent upon the feature geometry and location in the edge detected input image, were then minimized. When the search is started close to the actual feature location, the feature is found and its parts are classified well. If a gross search over the entire image is done, the algorithm will have a tendency to settle on local minimum, such as eyebrows even with a plain background. A complex background couldn't even be approached.

In [craw92], a larger number of facial features are classified than in [yuil92]. In addition, simple global constraints are imposed upon the geometry between features by use of model experts. Results are drastically better because local minimum can be largely identified by use of contextual constraints imposed by heuristics. As an example, it might be stated that the eyes must be detected above the mouth. Through the use of these global constraints, the system would not fall victim to local minimum on the face as would [yuil92]. Nonetheless, it could not handle a complex background.

The algorithm described in [yang94] is markedly different than the two aforementioned. The templates are less rigidly defined and global constraints are more strictly defined. Whole faces are searched for in low resolution images. Once a face is identified, features within it are searched for using edge based techniques similar to those described above. The algorithm depends strongly on the orientation of the head being upright and straight forward for the most part. Nonetheless, it performs robustly when forced to employ a gross search, even in complex backgrounds.

In [yow96], the deformable templates are defined in even less detail, but the constraints upon their relative orientation is better defined by use of a Bayesian network. Rotation of the face is allowed as long as all frontal features considered(eyes, nose, and mouth) are visible. Several bandpass filters are convolved with the image where the frequencies passed

are slightly different, filter to filter. Edges were searched for in each of the filtered images. The edge detected, filtered images were then searched for features. A great number of false positives were detected, but it was found that the true features were consistently identified in the same locations, whereas the false positives were more randomly scattered. By use of this insight and global constraints, features are identified quite well, regardless of the background and orientation of the features.

In [krug96], it can be seen that Gabor wavelets were employed to describe the specific features. A *jet* is defined as a set of Gabor wavelets where different frequency orientations in addition to different scales are represented. Elastic graph matching constrains the relative orientations of the features. A graph is required for each head rotation that is to be represented. The *jets* and graphs are trained on a large database of face images to account for differences not only in the jets themselves but also for differences in facial structure over people. Confidences are derived from the wavelet analysis that takes place at each feature which effect the overall structure of the graph. Thus, the overall structure and orientation of the head is estimated.

Although the algorithm [shok98] is designed for generic object recognition, it could be easily trained for face detection. Again a wavelet approach is employed. A *scale-space cell* or *SSC* is defined as an image segment that contains similar frequency content radiating outward from its center. To detect an *SSC*, the image is searched at multiple resolutions for image segments that maintain the frequency characteristics of a point. Saliency of the *SSC's* is based upon filtered local energy which is calculated for every (x, y) . The peaks in saliency form what is called a saliency map graph which is characteristic to the orientation of the features or *SSC's* detected. Graph matching techniques are employed to detect orientation in addition to drastic image deformation including feature occlusion.

It was implied above that schemes that maintained more sophisticated global constraints upon the allowed relative feature orientations provided vastly more robust detection when input images contained complex backgrounds. It should be noted, nonetheless, that it would be very difficult to implement systems such as these in real time. The most important thing

to be taken from the above analysis as it pertains to face detection is that relative feature orientation is an incredibly powerful constraint. As can be seen in [purc88], people tend to detect faces with some sort of similar constraint.

Holistic Systems

Holistic systems attempt to detect faces by use of representations that emphasize global nuances of the face and that do not necessarily correspond to features such as the eyes and nose. The simplest of these types of methods, template matching, is a traditional tool in pattern recognition [shan88]. Two other, more sophisticated approaches are those that employ neural networks and or eigen decomposition. The use of neural networks are known for their ability to estimate complex nonlinear functions, of which, face images are certainly a subset of. Eigen decomposition has been effective in linearly estimating nonlinear functions, such as faces.

Template matching is horribly fragile in varying lighting conditions. Nonetheless, given optimal conditions, real time operation is easy to achieve, if local area searches are incorporated. A system that relies strongly on template matching is seen in [mach96]. In addition to lighting, rotation and scaling are not deformations that are intrinsically handled. These problems were analyzed in [brun95], where methods for template matching and an incorporation of invariance to rotation and scaling is reviewed. Several approaches were presented where each design attempted to emphasize the importance of high frequency information. Real time operation was not discussed.

The neural network approach described in [rowl98] employs a receptive field structure. This structure was chosen to isolate horizontal stripes and different size blocks of the input image segment. It was shown that, if the results of multiple networks of identical structure and similar training were merged, the performance of the overall system increased greatly. Each network will differ markedly with another despite identical structure and similar training, due to the fact that the weights, before training, are set randomly and that there are many

solutions for the networks to converge upon. The images segments are largely pre-processed, to normalize for lighting and contrast problems.

The eigenface approach[turk91], a view based approach, decomposes a database of face images into eigenface space. Thus, each new image segment must be measured for *faceness*. *Faceness* is measured as the euclidean distance between the image segment and the image segment projected back from face space. When the image segment is projected back from face space, it is simply equal to a weighted sum of images in the face space. A weight vector describes the likeness of each prototype eigenface to that of the image segment under consideration. This system operates optimally with a large training set of faces. Some problems arise when lighting conditions differ between the training set and the input images. In addition, the system is highly sensitive to scale differences, so each image must be searched at several different scales. The search must take place over every (x, y) in the image at every scale. The location where the smallest distance is measured is deemed to be the location of the face. A more sophisticated but similar approach can be seen in [blac96].

Both of the above algorithms have trouble modeling lighting changes, and scale deformation. Therefore lighting normalization and scale searches are necessary. A system described in [sung94] uses similar technology but requires slightly less restrictive preprocessing. Similarly to [turk91] a face space is constructed using training faces. A vector is constructed, similar to the weight vector described above, that serves as input to the neural network structure. A similar algorithm is portrayed in [lawr97].

A positive aspect of the approach of these systems is that they intrinsically incorporate relative feature orientations into the detection operation. More easily than analytic schemes, holistic schemes can be implemented in real time. Unfortunately, as mentioned above these systems have difficulty with scale and lighting deformation.

Color Systems

Color schemes employ statistics pertaining to skin and feature tones in order to identify faces. This is the case in [oliv97], where head and mouth shape is accurately measured in quasi real-time. Heuristics, such as minimum expected size of the head, in addition to relative feature size, weed out false positives.

The authors of [dai96] relied upon the assumption that skin contains orange tones. They employed the *YIQ color model*, where the *I* channel represents the tonal values from orange to cyan. The lesser the values of the channel's pixels, the more orange and less cyan that exists in the image. Given this, any pixel in the *I* channel that contained a high value was set to zero. To distinguish between other body parts and the face heuristic methods are necessary.

An algorithm that is almost entirely based upon the use of color statistics is found in [sobo96]. The facial skin and features are segmented from the image and thereby detected. Some heuristics about geometry are used to eliminate grossly obvious errors.

In [waib96], a skin tone search is coupled with motion prediction and a few heuristics. Motion prediction is accomplished simply by estimating current position and velocity. This system operates for more than one person in addition to detecting faces at all orientations. It can be seen that, if skin is visible from other parts of the body, the algorithm may confuse that part of the body with another face.

It can be seen that, by nature of the approach, these algorithms are highly sensitive to lighting conditions. Their effectiveness should go down rapidly without maintaining consistent lighting conditions from training to operation. In addition, because they directly rely upon the image intensity values, a bright light within camera view can create a non-optimal use of quantization levels in the image. In other words, there may be many quantization levels that are set aside in the image that are not needed because the portions of the image that are not part of the light are relatively dark. This was highlighted in [brad98],

where an extensive review was given of the problems with color approaches to face tracking. Nonetheless, these systems can be implemented in a manner that will allow them to operate very quickly on conventional hardware. It should also be noted, that these systems cannot perform reliably without the use of heuristic tools, similar to those seen in analytic systems.

Conglomerate Detection Schemes

The systems described below incorporate several of the computer vision methods described above to achieve performance that is required for the particular application that the system is designed for. Generally, the systems are composed of a face detection algorithm that provides accurate estimates of the features but is computationally complex. Given that the feature estimates pass some sort of heuristic based test, a fast tracking algorithm is then employed to maintain feature estimates in real time.

In [jeba97], a skin tone search is coupled with a symmetry constraint in order to detect facial features. The orientation of the face is measured by use of perspective calculations. The texture map of the face is warped back so that viewpoint is at the front of the face. The eigenface system[turk91], is employed to verify that a face exists in the predicted location. If not, the original search is employed until a match is found. When a face is found, sum of squared differences is employed to track the features. In this system motion prediction is accomplished by use of an extended kalman filter.

In [zhan97], deformable templates are used to track mouth movement, and changes in orientation and translation. It maintains an operational speed of ten frames per second which is impressive considering the accuracy of the estimates of lip location that are provided. A slower algorithm that uses constraints relative to other feature location is employed to estimate the location of the mouth in the first frame [zhan96].

In [hein98], a two dimensional template matching scheme is employed to find features then a three dimensional system is initiated to track the features. It is noted that the framework

for face finding is very similar to that found in [jeba97] and [zhan97], but in this system all portions of the algorithm run in real time. The only manner by which the initial algorithm runs in real time is that it is implemented upon specialized hardware.

It should be noted that, if we use the human visual system as a model, current research suggests the importance of motion prediction. In [sinh94], it was shown that people actually integrate motion components over entire objects, instead of tracking just a few local features. So there must be some method for checking the consistency of relative motion estimates of each feature tracked. It should be understood that this is the relative feature orientation constraint considered temporally.

Systems That Employ Physical Apparatus

Systems of this type range from using computer vision to electro-magnetic fields to detect the human head. Irrespective of the detection method, they all require the person being tracked to wear some physical apparatus.

One of the most widely used systems for position detection is the Polhemus sensor [polh98] which uses electro-magnetic field measurements to detect position. Systems produced by [orig98] and [omni98] require stickers or small lights to be placed on the head to be tracked. They then employ standard vision techniques to extract the position of the head. It should be clear that systems of this type only track the small sticker or light and do not know the difference if it is attached to a head or any other object.

1.2.2 Face Rendering Systems

The desire is to create an output image that has the appearance of the face from an arbitrary viewpoint. Face rendering systems typically employ some sort of three dimensional model or utilize what are called image based rendering techniques. The approaches reviewed can be sorted into three groups; image based rendering algorithms, head model algorithms, and

model construction and deformation algorithms.

Image Based Rendering Algorithms

Image based rendering algorithms bypass the acquisition of a three dimensional model [shas96]. They require a set of correspondences between images in order to combine information to produce new views. The results are typically more realistic than traditional modeling algorithms.

A promising algorithm is presented in [seit96], where heads and other objects are rendered by use of morphing between two views. Morphing typically introduces deformations of the objects because the transformation is not constrained by the three dimensional geometry of the objects. To eliminate this problem, each view of the object is pre-warped from its orientation to an orientation that is parallel with the other views. It is understood that three dimensional geometry is not disturbed by morphing parallel views. To produce the rendered output at the desired orientation, the parallel views are morphed into the parallel view morph. The parallel view morph is post-warped to the final three dimensionally correct rendered output. It is claimed that the algorithm does not require a three dimensional model to perform this operation, but it does require many correspondences between the objects which is a similar type of information. Nonetheless, the algorithm produces very aesthetically appealing results.

In [beym95], both the texture and shape of faces are represented in vector form. The thrust of this algorithm is to be able to vectorize faces such that a particular index of a face vector references the same feature irrespective of which face the vector represents. Given two images of the same face at different orientations and a manner by which to represent the rotations to transform between them, that same rotation can be achieved for an arbitrary face. Because of the face vectorization, the rotation of the input face is simulated using its geometry, not the test set's geometry.

Head Model Algorithms

A generic three dimensional head model is employed in the following algorithms. Synthetic rotations of a head can be simulated simply by rotating the head model from one orientation to another. It should be noted that these algorithms approximate the structure of each head by use of the generic head models.

In [vett96], the generic head model is employed for warping the original images to the virtual view. The attempt is to separate shape processing from texture processing. By this manner, holes in the warped input texture maps can be filled with a linear combination of test textures. In addition the shape of the input head can be approximated by the use of a linear combination of the test shapes. Provided the rotations are represented within the test set, convincing morphs are produced.

In the system described in [eise97], texture information is maintained in advance, and action parameters are extracted from video. This way the face is actually animated not rendered frame to frame. This requires a reasonably detailed deformable head model in addition to a detailed facial action detector. As a result the detection does not operate in real time. Nonetheless, rendering results are surprisingly good. A system shown in [mcco97] is similar, but incorporates a recently designed face tracking system [mach96].

It should be understood, that the results of these algorithms cannot compete with those produced by the image based rendering algorithms. Unfortunately, the image based rendering algorithms require far more correspondences than a face detection or tracking algorithm can provide in real time. The head model algorithms simply need enough features detected that pose of the head can be recovered, namely four [horn86].

Model Construction and Deformability Algorithms

The algorithms described below should produce the best results of the three different types. Unfortunately, to gain the positive effects the computational complexity is ridiculously high.

In [pede96], construction of a virtual view of a face was produced by calculation of the orientation of normals to the face surface. In other words, the three dimensional geometry of the specific face is measured. Three calibrated cameras are required. The output model is very smooth, but convincing once texture mapped.

In [izqu97], stereo matching was employed to extract three dimensional information about the geometry of the person in view. The disparity maps derived from the matching are used to interpolate the views. Large orientation changes aren't represented, but results for those that are represented, are aesthetically pleasing. It should be noted that the work only considers interpolations between the camera views held.

A system that deforms a generic three dimensional model was presented in [deca98]. The deformation is based upon many defined feature locations. Certainly the number of feature locations that are necessary could not be identified or even tracked in real time. It would be possible to identify these features over time, in order to slowly cater the model to a particular user's head.

It also should be noted that in addition to vision based systems there are several systems that use other sensing media in order to develop three dimension models of non-convex shapes. The most well known system such as this is the Cyberware system [cybe98]. In addition, Polhemus has developed a *Handheld Laser Scanner* [polh98].

1.3 System Overview

The system presented in this work has two main functions; face finding and face rendering. The face finding algorithm can be further classified into three major algorithms; face detection, face validation and face tracking. A similar overall framework for face finding is seen in [jeba97] and [hein98]. The face rendering scheme loosely resembles work in described in [seit96], where extra steps are employed to render in three dimensions without distortion. A general system diagram can be seen in Figure 1.3 and a picture of a representative physical setup is seen in Figure 1.4.

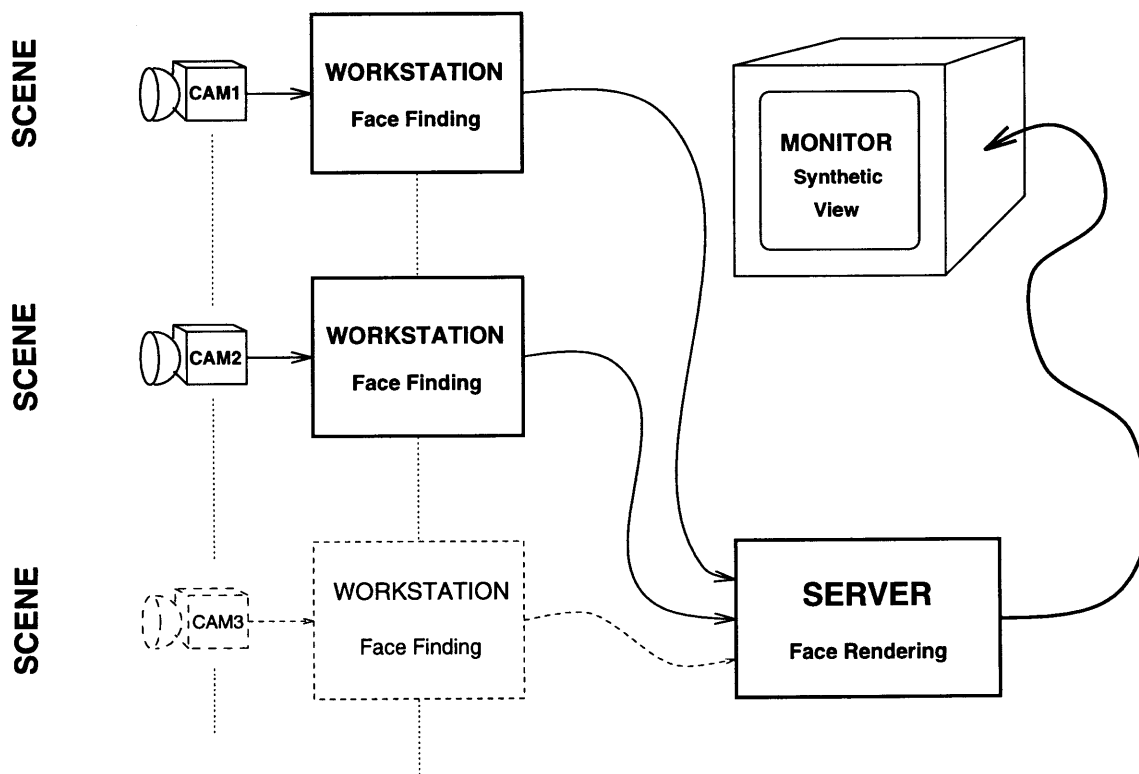


Figure 1.3: The cameras on the left view the scene. Their relative orientation is unimportant for operation, although there are orientations which result in better system performance. The live video is scrutinized by the face finder in each workstation, where the number of camera workstation pairs is scalable. The texture maps and feature locations are sent to the server to be composited. The server renders the virtual view and outputs the results to a display device(monitor).

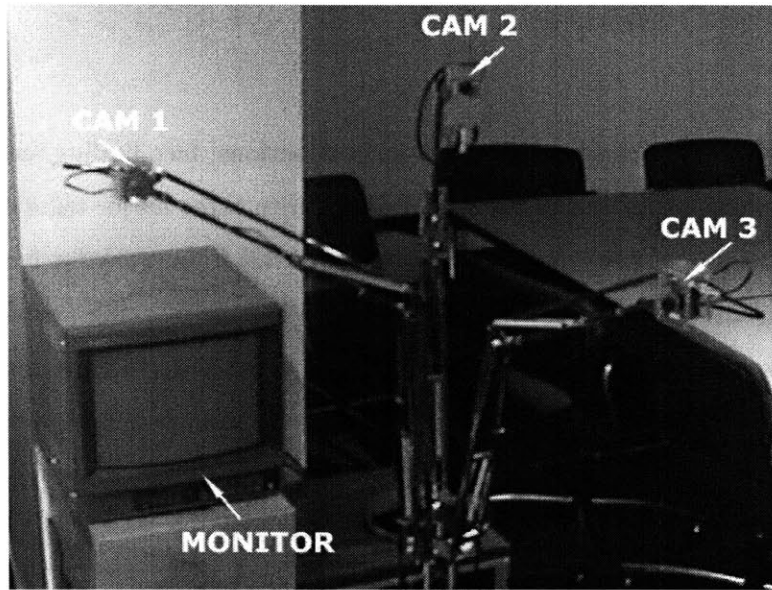


Figure 1.4: As can be seen in the picture above, the cameras are dispersed so as to maintain significantly different views from one another. It should be understood that this is only one of many suitable camera configurations.

1.3.1 Overview of Face Finding

It can be seen in Figure 1.3 that the face finding algorithm, running on a workstation, operates upon live video. Although there are three algorithms contained in the face finding system, only one will be running at one time.

The face detection scheme operates when there is no current set of feature estimates for the last frame. Intuitively, this condition is true either when the system is first started or when the face had been tracked by other means and been lost. The algorithm searches the entire input image for facial features which, as stated before, are defined as the eyes, nose, mouth and ears. Although the quantification of the human face is outside the focus of this work, it is necessary to identify some important visual tautologies for recognition. I assume that the tonal values of skin do not change drastically from subject to subject, that features are typically darker than non-features, and that features contain more edges than non-features. Nonetheless, the system described herein requires only a majority of the heuristic constraints to be satisfied in order for recognition to take place.

The face validation routine is engaged when the face detection routine has developed a set of potential feature locations. It is responsible for declaring whether or not the features that were detected belong to a face, thereby measuring the *faceness* [turk91]. If the image segment doesn't contain a face, the face detection algorithm will be called again to refine its estimates.

The face tracking algorithm engages when the face validation routine declares that a face exists at the feature locations found by the face detection system. Templates are extracted and normalized correlation is used to track the features. The system is real time but somewhat sensitive to drastic changes in lighting, occlusion, and vast changes in feature location frame to frame. In the event that the tracker loses features, the face validation routine is initiated to check if there is still a face at the current location estimates. If there is no face present, the face detection routine is initiated. If there is a face present, tracking can continue, although, later we will suggest further action, in this instance.

1.3.2 Overview of Face Rendering

It can be seen in Figure 1.3, that the face rendering system operates on the server in the system. A generic three dimensional head model is employed to maintain estimates of the structure of the detected head and face. Given that the cameras in the system have sufficient coverage of the entire face, every viewpoint can be rendered.

In order to satisfy the real time constraint, in addition to providing reasonable accuracy, rotations are simulated by two dimensional warp maps. Each map can only represent a rotation from one head orientation to another. Because each warp map is an integer image, the memory requirements of the warping algorithm would be staggering. It behooves us to maintain pre-warp maps that warp from arbitrary head orientation to some zeroed orientation, and post-warp maps that warp from the zeroed orientation to the output orientation.

In optimal operation, the system will detect and estimate the orientation of human heads at arbitrary orientation and render virtual views at any other arbitrary orientation.

Chapter 2

Face Finding

The face finding algorithm was composed in accordance to a design philosophy called the *workshop metaphor*, [ap96]. The philosophy calls for a structure to problem solving, similar to that evident in a workshop, where a hierarchy of decision making exists from craftman to manager.

As described in [ap96], several different types of craftsmen can accomplish a particular goal without help from one of a different ilk. The problem is that each of the capable craftsmen will only operate efficiently for a portion of work involved in producing the desired result. The *workshop metaphor* describes the use of several capable craftsmen with a manager that understands the strengths and weaknesses of each craftsmen. This way the manager finds a way to structure the work so that no craftsman is forced to operate inefficiently. Consequently the choice of which manager to use is an important one.

This approach to system organization is similar to the algorithms that were described in the first chapter as conglomerate algorithms, where several computer vision techniques are employed for recognition in order to achieve performance that would be impossible using only one [hein98], [jeba97], [zhan97].

2.1 System Description

As described in the previous chapter the face finding system is composed of three major algorithms; a face detection algorithm, a face validation algorithm, and a face tracking algorithm. Each of these algorithms operates as a craftsman with its own set of strengths and weaknesses. Given the goal of producing real time estimates of feature locations the manager must decide which of the three algorithms to initiate at what time. Thus it is important to give a general review of the characteristics of each of the algorithms. An overview of the system can be seen in the flow chart in Figure 2.1.

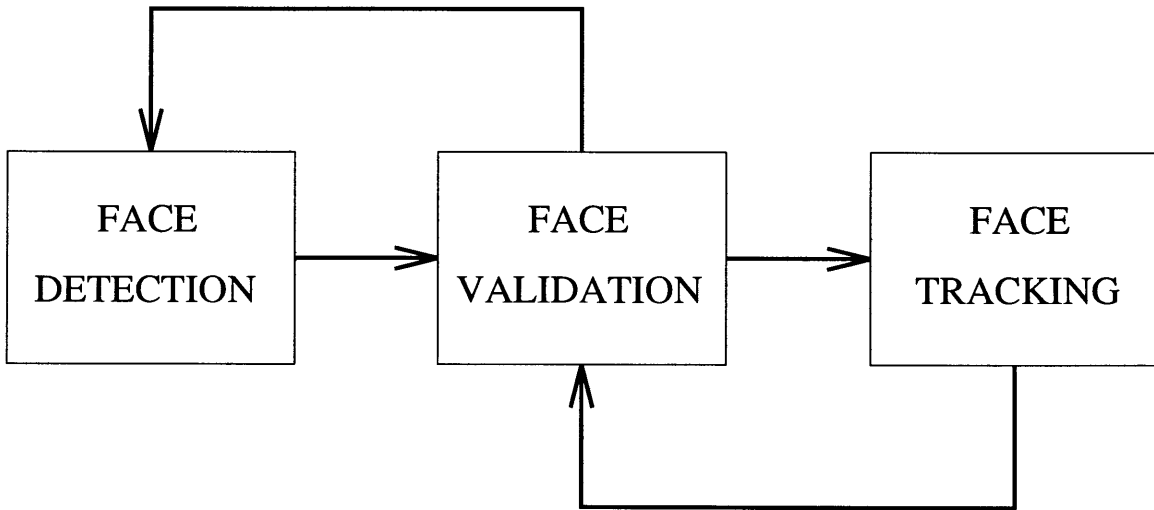


Figure 2.1: The general flow from one algorithm to another can be understood from the diagram above. The face detection algorithm finds facial features. The face validation algorithm decides if the features that were found belong to a face. The face tracking algorithm tracks the features found in real time.

The face detection algorithm detects facial features at a wide range of orientations, scales, and lighting conditions. Nonetheless, it operates at about three frames per second and does not contain any constraints based upon relative feature orientation and thus detects a significant number of false positives. Because of the speed of the algorithm it cannot be operating for more than a few frames every now and then.

The face validation algorithm produces accurate estimates of the *faceness* of image segments and intrinsically constrains features based upon relative location. Nonetheless it is very

sensitive to the scale and orientation of the face.

The face tracking algorithm tracks features given changes in orientation and scale and operates at thirty frames per second. Unfortunately, it is very sensitive to changes in lighting conditions and drastic changes in feature location frame to frame.

Each of these algorithms could be optimized to find faces alone. Unfortunately though, based upon their intrinsic weaknesses, they would accomplish the goal at the sacrifice of accuracy, speed, and elegance. Each algorithm would be forced to operate beyond its range of efficiency.

Because each algorithm's strengths and weaknesses can be quantitatively explained, the simplest management approach is to maintain a three state state machine.

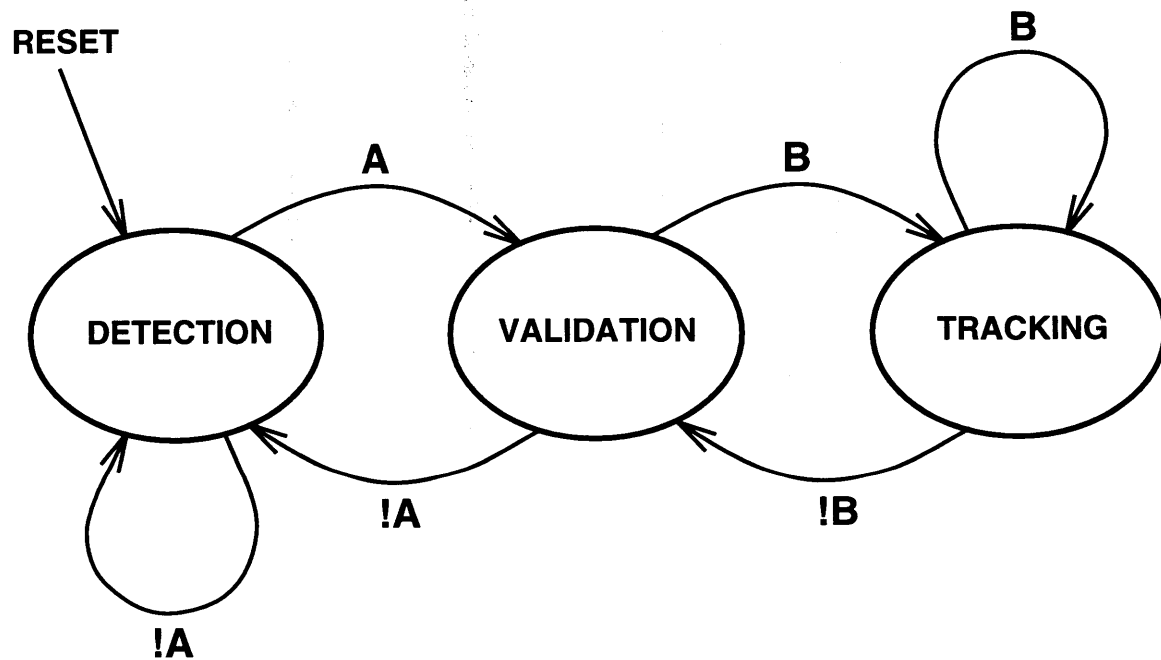


Figure 2.2: The state diagram above highlights the decision making of the management of the algorithms in the face finder. Condition **A**, takes place when feature estimates are produced and condition **B** takes place when features are validated as true. Naturally the **!** symbol indicates the opposite of whatever it precedes. It can be seen that the detection algorithm is initiated when the system is first started, and when **!A** is true. The validation algorithm is initiated given either **A** or **!B** is true. The tracking algorithm is initiated when **B** is true.

As can be seen in Figure 2.2, the face detection algorithm is initiated when the system is first brought up, or when there are less than four accurate feature estimates. It will operate until it detects at least four prospective feature locations, at which time the face validation algorithm will be initiated. Face validation simply produces a rating of *faceness* of the image segment. Based upon this rating the manager will declare whether or not a face is present in the image segment under consideration. If there no face present the face detection algorithm is initiated again. If there is a face present, the tracking algorithm is initiated. This algorithm will operate indefinitely, until it does not have a reliable estimate of at least four features, at which time the face validation routine is initiated.

2.1.1 Face Detection

As has been stated above, the face detection scheme is computationally intensive but provides accurate estimates of human facial features. The algorithm is independent of head orientation along all three axis of rotation (X,Y,Z). It is based upon three major observations: skin tone is similar across people[brad98], features typically have darker tonal values than non-features, and features have edges[shok98].

As each frame is digitized, a skin tone search is conducted within the image. A multiple of the variance of the pixels in a skin tone sample is used to open an elliptical window of acceptable skin tone pixel values, centered around the sample's pixel mean, thus the skin tones are normally distributed. Skin tones blobs of significant size are grown to produce a bounding box using a nearest neighbor growth algorithm[horn86]. It is assumed that there is only one face within view at a time. Although at the sacrifice of speed and some accuracy, the system could be scaled to search for more than one face. If there are no blobs of significant size present in the image, the bounding box is set to the size of the image. Thus, if there is a face present, but it didn't show up using skin tone, it will be found using the other methods. Although unlikely, a face might not be detected using skin tone because of problems with lighting, makeup, or glasses.

Once a bounding box is set, the image portion contained within it is segmented from the frame. Dark areas of this image segment are highlighted and small blobs are grown. Blobs that become feature estimates are loosely constrained to be of appropriate size given the size of the bounding box. Thus the algorithm looks for smaller dark blobs that are surrounded by lighter tones. The estimates are stored over several frames.

Edges are then searched for in the image segment. The edge-detection algorithm used is a simplification of the Marr-Hildreth scheme. A Laplacian pyramid of the image segment is constructed. Locations where there are zero-crossings in the pyramid are deemed edges. Naturally the smaller pyramid segments tend to contain cohesive feature blobs but do not relate their locations very accurately. Larger pyramid segments maintain accurate locations of the features but tend to not be completely connected. Thus, a coarse to fine search is logical. Each edge is grown and checked for appropriate size starting with the coarsest pyramid segment and finishing with the finest. As with the feature tone search, the feature location estimates are stored over several frames.

Several images are pictured in Figure 2.3, that display a representative set feature estimates over several frames of the locations of the features. Given that the feature locations didn't change in the time it took to process the frames, there should be many *hits* in the feature locations. When a threshold for the number of *hits* is exceeded in four feature locations the orientation of the head is calculated. This threshold is certainly dependent upon the number of frames that contribute to the estimation. The head nor the camera configuration is constrained to be of specific orientation or reference location. Consequently, the head orientation must be checked for each potential feature configuration. Some of these configurations are physically impossible and can be ruled out right away.

There is no artificial limit on the number of total *hits* allowed per image segment. If more than four feature are detected, the system will select those that have the maximum number of *hits*. The face detection scheme processes at a rate of roughly three frames per second.

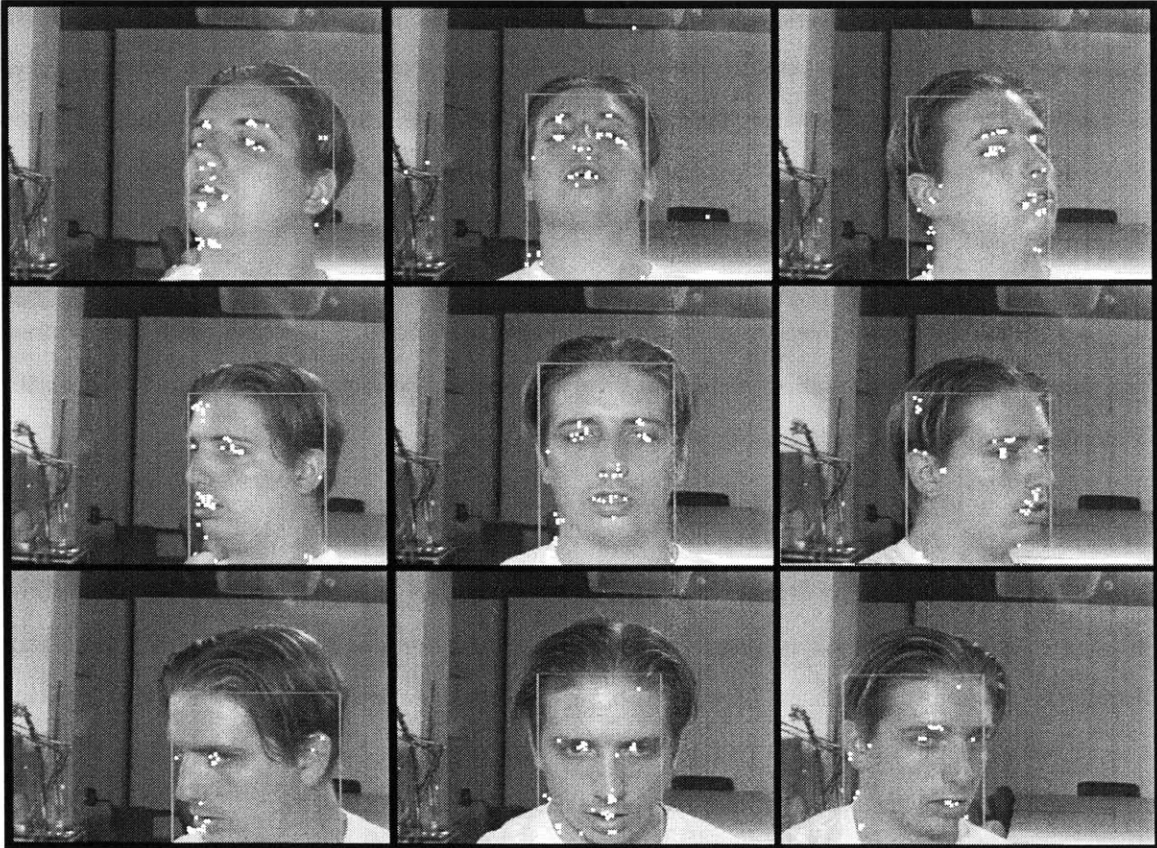


Figure 2.3: The white dots in the above images are locations in each image that appeared to have feature like edges and colors. The bounding box around the head was found by use of skin tone analysis.

2.1.2 Estimating Head Orientation

The orientation of the detected head must be accurately determined for several reasons. The prospective face must be warped to a zeroed orientation to measure the *faceness* of the image region. Provided the image region does contain a face, it must be known what orientation the head is at when the templates are cropped from the texture map in order to robustly track.

It requires four detected features to calculate a unique orientation of a rigid three dimensional object, given the corresponding features represented in three dimensions upon a model of the object[horn86]. If the features are thought to be the eyes, mouth, and nose, the angular displacement of the detected face to the head model is measured from a mug shot of the face. If the features are thought to be an ear, eye, mouth and nose, the angular displacement is measured from a profile shot and then corrected later. It was found this scheme provided more accurate measurements due to slight errors in feature estimates and differences between the head model and the face being detected. Nonetheless, for the purposes of this work it is assumed that the three dimensional structure of every head and face detected by the system can be roughly approximated by a three dimensional head model as seen in Figure 2.4.

Being that it is a relatively fast operation, a weak perspective calculation is done for each feature configuration[jeba97], [alte92]. A particularly simple and fast algorithm was presented in [alte92]. A detailed review of other algorithms was also present. This calculation provides a measurement of rotation about both the **X** and **Y** axis. The **Z** axis rotation can be measured assuming that it accounts for all angular displacement once the **X** and **Y** axis rotations are corrected.

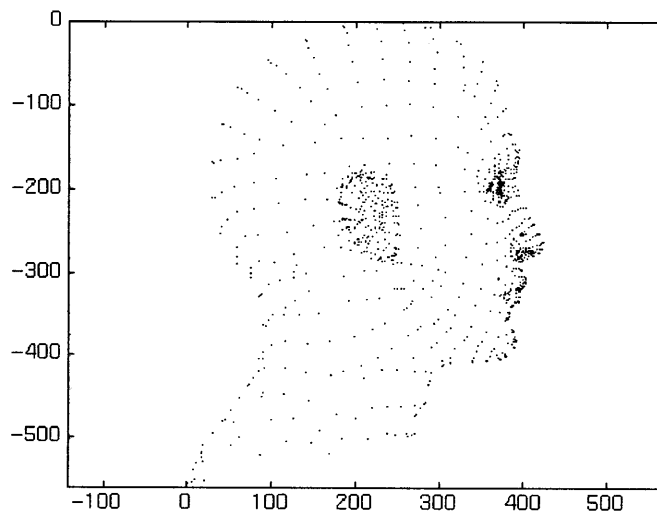
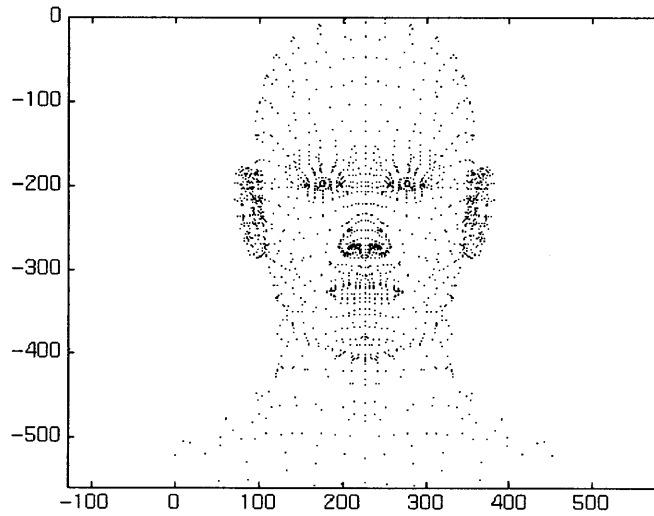


Figure 2.4: This the head model employed for the online measurement of head orientation. It is also used to generate the warp maps that alter the viewpoint of the face. The **X** axis can be thought to run from the left ear to the right and the **Y** axis can be thought to run from the bottom of the neck up through the top of the head. Naturally the **Z** axis would then be extending from the back of the head out through the nose. The model was created by Viewpoint Datalabs[view98].

2.1.3 Face Validation

Upon detecting a set of facial features, and having calculated the corresponding head orientation, the algorithm must decide if the set of features detected belong to a human head. Several algorithms perform this function as discussed in chapter one [rowl98], [turk91], [sung94], [lawr97]. The eigenface algorithm [turk91] was selected due to ease of implementation, although any algorithm that provides a reliable estimate of the *faceness* of an image segment will suffice.

As stated before, the system must be able to detect faces of any orientation. Given that the eigenface system is very sensitive to translation, scale and rotation, it is necessary to provide the system with a consistent facial view. This highlights the necessity of rendering the face at a different orientation than is detected in the camera and then proceeding to validate it. Which orientation is appropriate for validation?

It is logical that a profile shot does not contain enough information pertaining to the front view to accurately measure its *faceness* from the front. Similarly, there is not enough information in a mug shot about the side of the face to measure *faceness* from the side. Moreover, it is not desirable to have to check each feature configuration for both *front faceness* and *side faceness* because of computational complexity and inelegance. In addition, problems would be experienced if the head orientation is halfway between a profile or a mug shot.

Measuring the distance between an image segment and face-space reduces to simply a pixel by pixel multiplication and accumulation. Given this we can consider isolated portions of eigenimages if needed. So, we need to determine which pixels in the image are the ones we want to project into eigenspace.

Given that we know at least four feature locations, and the head's proposed orientation we can warp the texture map from its orientation to a different one. The texture maps are warped to an unwrapped template of the head, whereby rotation about the X and Z axis

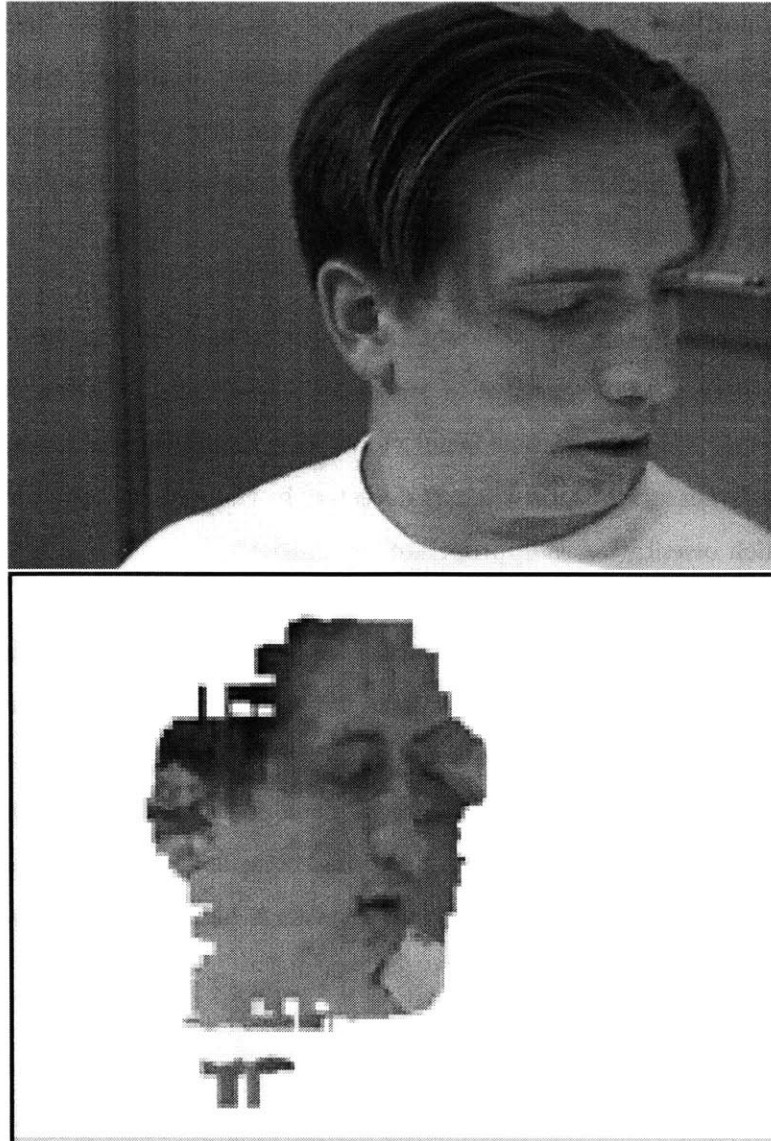


Figure 2.5: The top image in this figure is the original image. The image below shows the face warped from its original orientation to the base orientation. Confidence maps will be used to crop the erroneous data away from the warped face.

are nulled and rotation about the Y axis has been transformed into horizontal translation across the destination image. A representative video frame can be seen in Figure 2.5, where the face is found and warped to the base orientation. The destination image is referred to as the base image. The warp maps are fabricated from the three dimensional head model as seen in Figure 2.4, which is rotated to simulate the orientations that human heads can be found at.

Warp maps that account for the transformation from each angle of orientation to the base map must be fabricated. Confidence maps decide which image portions that should be projected onto eigen-space. Without their use, many non-face portions of the image would also be projected, thus rendering the results useless. By providing several training images from different views of a each person, a complete base texture map can be composed by use of the warp and confidence maps. An indepth discussion of the production of warp and confidence maps will be given in the next section.

It must be noted that it is important the warp maps do not intrinsically filter out the high frequency information in the faces. It has been noted that high frequency information is a key component in recognition[brun95].

2.1.4 Tracking with Warped Templates

Upon the detection and verification of the feature locations, templates of the features can be extracted for tracking. Normalized correlation is employed to track features in two dimensions. A two dimensional scheme is desired due to processing requirements. Real time operation is a solid constraint.

As template matching is a computationally intense task, local area searches are done. In other words, the matching algorithm employed only searches until a confidence metric surpasses a threshold or exhaustively searches a range proportional to the original size of the head as measured by the skin tone bounding box. If the range designated is exhaustively

searched, it can be assumed that the feature has been lost.

The formula for normalized correlation (NC) is given by,

$$NC = \sum_{x,y} \frac{(I(x,y) - I_m)(T(x,y) - T_m)}{(I(x,y) - I_m)^2(T(x,y) - T_m)^2}$$

We would like to maintain the speed of a two dimensional tracking scheme while gaining the results of a three dimensional system. We will accommodate for this by using a similar approach as depicted in the face validation section but with one extension. We would like to warp the extracted template for each feature to a base template image. Then we can use the priority maps to composite a final template. As with before, rotation about the X and Z axis is nulled and rotation about the Y axis has been transformed into horizontal translation across the base template. From this base template, templates of every orientation necessary can be post warped. Each face tracking algorithm, running upon each workstation, should maintain the same full set of templates.

The head orientation calculated per frame should be used to decide which templates to use. Template matching systems such as this typically employ an estimation algorithm to generate more robust estimates and to constrain the estimates based upon mutual information. An extended Kalman filter will suffice, although it should be noted that there should only be one filter not one filter per camera.

2.2 Learning a Face

Template matching is a very fast and efficient method for tracking features on a rigid object. A problem is encountered when the object deforms, such as would be the case if someone being tracked smiles or closes their eyes.

The occurrence of this would cause the tracking system to search its entire allowed area.

Then the face validation routine would be called to see if a face still exists at the location the face tracker thought it did at the previous frame. If the face validation routine is aptly trained it will declare a that a face exists and as usual templates will be extracted and the face can continue to be tracked. By simply saving the past template sets, the algorithm acts as simple but automatic unsupervised learning algorithm. A similar approach to unsupervised learning can be seen in [darr93].

Although this is not the focus of this thesis, this information is valuable for automatically deforming the three dimensional model used for warping. By this manner, vastly more accurate rendering results would be achieved. Nonetheless this topic is outside the scope of this work.

Chapter 3

Face Rendering

The goal of this rendering system is to accurately render the human head from an arbitrary viewpoint in real time. Three dimensional polygon rendering systems are common for these applications, but cannot be implemented in real time without specialized hardware. So rotations are simulated with warp maps. In other words, lack of processing speed is compensated for by use of larger amounts of memory. Results should be of similar quality, given memory.

3.1 System Description

Once the system has found the facial features, declared them as belonging to a face, and calculated the orientation of the face, the system is prepared to render the head at an arbitrary orientation. It should be noted that this output orientation is determined by the application. If the cameras are placed in a way to provide maximum coverage of the face, results will obviously be better. Nonetheless, no requirements are made upon the placement of the cameras. The challenges posed are to be able to warp without losing accuracy, without over spending memory, and to merge results in an aesthetically appealing manner from all views.

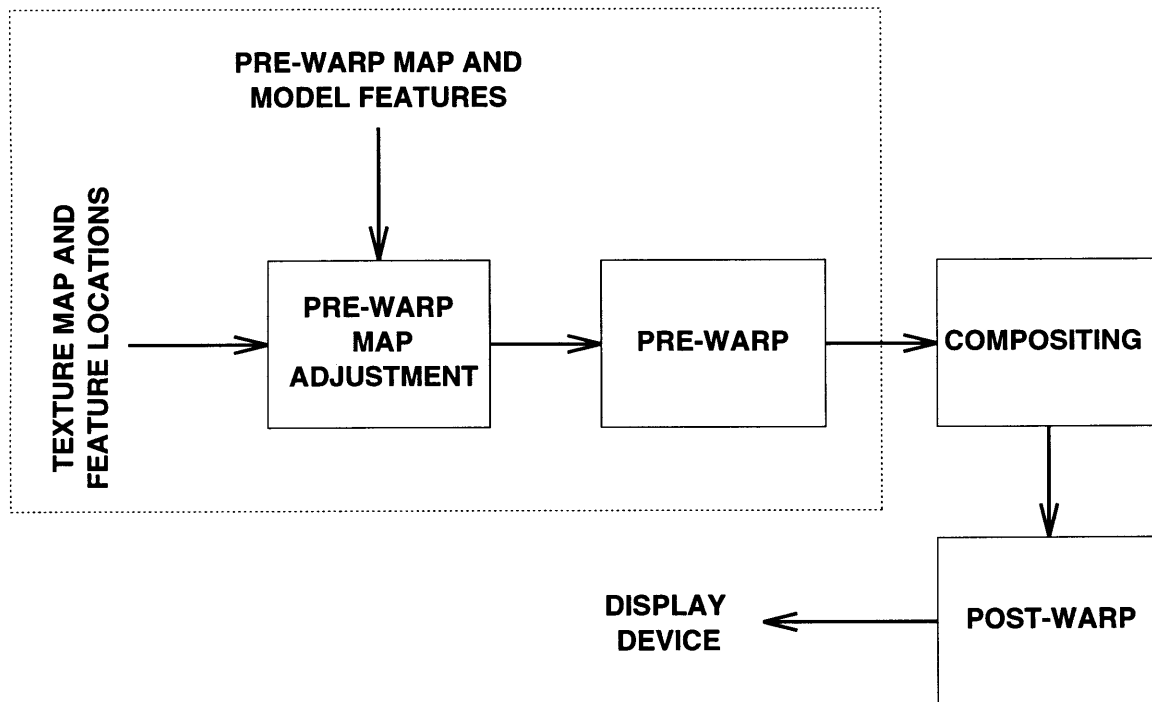


Figure 3.1: It can be seen above that the pre-warp maps are adjusted to geometrically fit the incoming texture map, based upon the known feature locations. Texture maps from each camera view are then pre-warped to the base orientation where they are composited with one another. After the final base map is composited, it is post-warped to the output virtual view. It should be noted that the portion of the system that is included within the large dotted box is scalable, and in this implementation there are three of those such blocks.

An overview of the face rendering system can be seen in Figure 3.1. Each texture map, at arbitrary orientation, is warped to a base image. The base images from all the camera views are composited into one base image by use of confidence maps. This final base map is then post warped to the desired output orientation.

3.1.1 Warp Details

The goal is to render heads at any orientation given texture maps of heads at any orientation. Because each warp map represents only one rotation, it would require 1000^2 (614.4 Gigabytes) warp maps per start orientation! This calculation is based upon only providing warp maps at every 20 degrees along all three axis, where each warp map (a map for \mathbf{X} translation and \mathbf{Y} translation is necessary) is realized by a (320x240) array of integers. This is obviously not feasible. A pre-warp and post-warp system is required.

Another problem in rendering faces is deciding upon the method for compositing the different texture maps. Differences in lighting and cameras can introduce images that although appear fine, contain texture maps of the face that are drastically different from one another. Some sort of filtering of the texture maps where they meet is necessary to maintain a smooth transition. By use of the pre-warp and post-warp system we implicitly merge the texture maps where by no extra filtering is necessary. A system such as this thus streamlines the merging of multiple view information.

Pre-warp maps are constructed by rotating the head model from the base orientation to the orientation that the face is detected at. It should be understood that the model at the original base orientation is unwrapped to produce the final destinations in the base map. This is done simply by centering the model around the \mathbf{Y} axis and mapping the angular displacement around the \mathbf{Y} axis of each model point to the horizontal dimension in the warp map. Thus the texture map is warped directly from its inherent orientation to the base orientation where \mathbf{X} and \mathbf{Z} rotation is zeroed as shown in Figure 3.2 and \mathbf{Y} rotation corresponds to translation in the horizontal dimension.

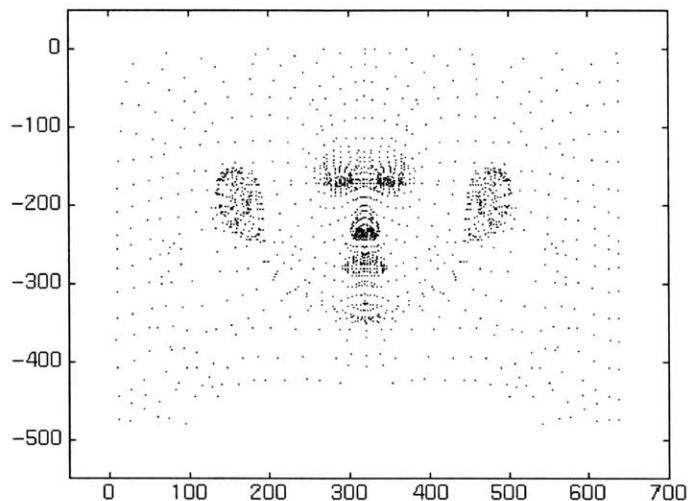


Figure 3.2: The destinations of the texture maps can be inferred from the model above based upon where the features are. It is this orientation where \mathbf{X} and \mathbf{Z} axis rotation are considered zero, and the \mathbf{Y} axis rotation is considered horizontal translation.

There must be some sort of hidden surface removal algorithm such that portions of the model that correspond to regions of the head that are occluded are not included in the production of the warp maps. Outward surface normals are given for every point in the head model, so performing hidden surface removal is made much easier. Any point's surface normal that incurs an angle of ninety degrees or more with the optical axis is eliminated from the dataset.

It is understood that, with the warp maps derived to this point, only the pixels which fall upon the model points will be warped so some sort of interpolation is required. Delaunay triangulation, [shew96], is performed upon the known values, and a polygon mesh is constructed. The surface is then smoothed with a gaussian filter to remove the sharp edges between polygons.

The model feature locations must also be scaled and translated to correspond to the appropriate locations in the warp maps and then saved along with them. They will be used for online adjustment of the warp maps.

Warp maps that warp from the base image to the virtual viewpoint are fabricated in exactly the opposite manner. The post-warp maps will require no adjustment because it is assumed that the base map contains the texture maps appropriately aligned by the pre-warping algorithm.

3.1.2 Confidence Maps

Every base image is capable of displaying every bit of area of the head detected. Obviously any one camera view will not provide all of the information. So we need some way to composite all the camera views into one base image. To accomplish this, a confidence map is constructed for each warp map.

Each texture map of the head provides the most detailed information at the locations upon the face where the normal of the face surface is parallel with the optical axis. As the normals of the face incur larger and larger angles with the optical axis, the resolution of the face surface at that point falls off. We know which orientation the head is at, and we have an estimate of the orientation of the outward normals associated with it (from the head model). Thus a confidence map can be derived such that locations in the image where the normals are parallel or almost parallel to and in the direction of the optical axis are given high priority. Naturally, locations in the image where the normals are close to ninety degrees are given almost zero priority. The indices of confidence maps are used as a per pixel weighting function for compositing the final base image. The confidence maps are produced by use of the following equation.

$$cmap(x, y) = \frac{(255)2}{\pi} \cdot \arccos \left(\frac{\pi}{2} - \mathbf{S}_n(\mathbf{x}, \mathbf{y}) \cdot \mathbf{Z}_n \right)$$

$\mathbf{S}_n(\mathbf{x}, \mathbf{y})$ is the unit normal vector to the surface of the model that is angularly aligned with the detected face. \mathbf{Z}_n is a unit vector in the direction of the \mathbf{Z} axis. Naturally only those normals that incur angles with \mathbf{Z}_n less than ninety degree's are considered. The use of

the warp and confidence maps alleviates the need for multiple validations as was discussed above. A representative confidence map is displayed in Figure 3.3.



Figure 3.3: The image above displays the form that the confidence maps take. This map specifically was generated for the rotation of $(X_{rot} = -20, Y_{rot} = 30, Z_{rot} = 0)$. These rotations correspond to looking down twenty degrees, left thirty degrees, without twisting the head to the left or right at all. The erroneous data on the right hand side of the above image is not high enough in magnitude to dramatically effect results.

The composition of two texture maps can be seen in Figure 3.4. The faces are found within each view, warped to the base orientation and composited using the confidence maps.

It can be seen that depending upon the orientations of the cameras and with respect to the head being tracked, all portions of the face may not be visible at all times. Given that we have a confidence map for each of the views, we can composite them to produce a confidence map for the entire final base map. In this way we could either fill in the holes as in [vett96], or simply refuse to render it. Whichever output is decided upon, the point is that we maintain this information such that it can be used depending upon the application.

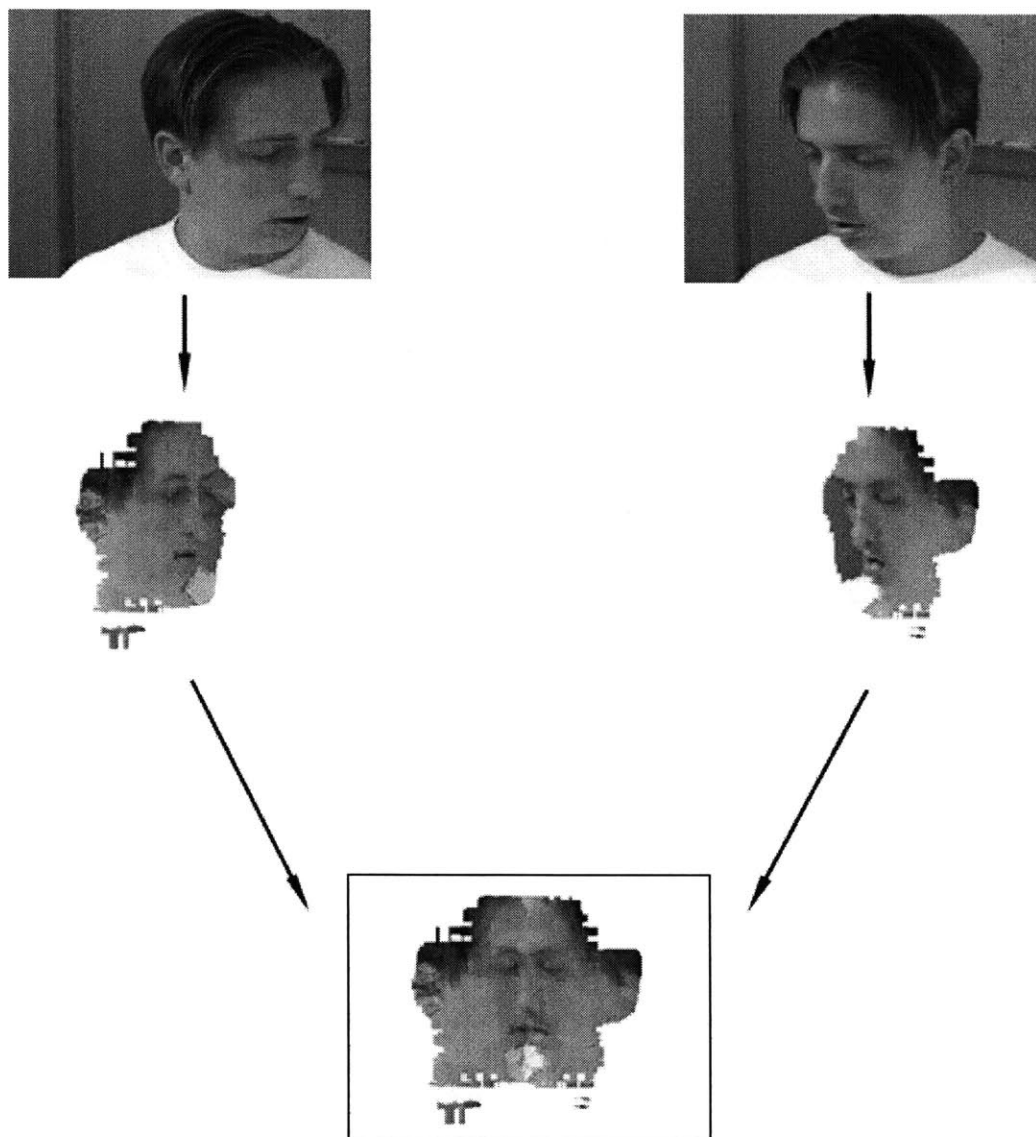


Figure 3.4: The input images above are searched for the each of the faces in view. The orientation of the heads is estimated at $(X_{rot} = -20, Y_{rot} = 30, Z_{rot} = 0)$ and $(X_{rot} = -20, Y_{rot} = -20, Z_{rot} = 0)$ respectively from left to right. The texture maps of each head are warped to a base image, whereafter they are composited together using the confidence maps. Notice that the by use of the confidence maps, correct facial data was chosen over the erroneous data when there was facial data available.

3.1.3 Online Adjustment of Maps

The pre-warp maps, as discussed above, warp texture maps from different arbitrary orientations to the base orientation. It is understood that there are regions in the warp maps that correspond to the features in the model. These correspondences must be transformed such that these regions correspond to features in the detected face.

This is accomplished simply by translating and scaling each cell in the warp map. Although it is beyond the scope of this work, it would be convenient to be able to also adjust the warp maps for differences between the geometry of the head detected and the head model [deca98].

Chapter 4

Implementation Specifics and Future Work

The system operates using one workstation for each face finder, in addition to one workstation for the rendering of the final output. The reason for the resources required is simply because mass-produced hardware is not very well optimized to run these types of algorithms. It should be understood that using specialized hardware for only one or two portions of the system would greatly lessen the cost of the system.

4.1 Hardware

The physical hardware required for the operation of this system is listed as follows:

- Three - Digital Personal Workstation 500au.
- One - Digital Personal Workstation 500.
- Four - Digital MME Cards, video digitizing.
- Three - Ultrack Video Camera, standard visible range, no filters.

- One - Sony Trinitron Monitor

The three DEC 500au's are dedicated to finding faces within each of the three camera views. Video is digitized on each machine by use of an MME card. The DEC 500 runs the face rendering system and outputs the rendered head to the monitor using an MME card.

It is necessary to work with video in the YUV format(4:2:2) because this is all that is supported by the MME video digitizer. If a conversion of video representation is performed it will do nothing but slow the system without need.

4.2 Software

The code for most of the basic functions was written in C. The function calls in addition to some pre-processing was written in Isis [agam97b]. Isis is a scripting language that allows for real time operation, and thus suited the application perfectly. Some basic and fast image processing functions were provided in a library and were used when appropriate. All networking, interaction with the DEC MME card, and file I/O was handled by the Isis libraries. It should be noted that some of the eigen face code was written in Matlab in order to utilize standard library functions in that software package. Naturally due the speed of Matlab, this code was targeted for pre-processing.

The use of Triangle [shew96], for computation of two dimensional polygon meshes, is gratefully acknowledged.

4.3 Real Time Considerations

In many cases throughout the system it was necessary to sacrifice other resources, such as image quality and accuracy, in order to achieve real time performance. In some cases, decimated video was used in-order to limit search time. In others, real time operation is

assumed only most of the time. It is necessary to outline the design issues surrounding the real time constraint.

In skin tone detection, the video was coarsely decimated to half its original dimensions in order to limit the time it takes to grow the blob of skin tones within it.

It was previously expressed that the template matching algorithm runs at a rate of thirty frames per second and employs a local area search in order to do so. It should be understood that under normal operation this scheme operates at thirty frames per second. If the system loses a feature it attempts to look a bit further for it and thereby takes a bit more time to operate. The design relies upon the idea that features won't be lost frequently.

As previously discussed the real time rendering of faces was achieved using warp maps. Because each warp map can only represent a rotation from one orientation to one other orientation, it is necessary to maintain many of them in memory. So memory was sacrificed for speed. Naturally warp maps cannot be maintained for every angle, so only those at ten degree increments are used. When the face is at an orientation in between two maps, the closest one is chosen. It was found that a disturbing amount of error was not introduced into the images despite this seemingly drastic approximation.

Traditional mass-produced hardware is badly configured for this type of system as shown in [hein98]. Specialized hardware can cheaply perform the sections of the algorithm that are the most computationally intensive. This would greatly reduce the cost of the end system.

4.4 Future Work

In the composition of the featured system and review of other work, it was seen that finding faces in real time can be assumed given constrained situations. These constrained situations would be the following: the lighting should not ever change, the face should not ever move very quickly, once within view the person shouldn't look away. Even these constraints can

be accommodated in different system designs, but the point is that real time face finding has become pragmatically possible.

Work seen in[breg97], a data driven approach, provided very accurate renderings of humans while speaking, although they could not render from any direction other than that which was provided in a database. The system required input from users and also a provided audio channel and was therefore not very automatic. Nonetheless, the rendered results serves as a good example of what the end goal in designing automatic face rendering systems should be.

To accomplish this goal, we must improve the art in three different areas. Our real time face detection systems should accurately detect the nuances of facial motion and structure. The deformation of the models such that the individual specific physical nuances, such as hair style and facial structure, can be accurately rendered from any view. The segmentation of the texture maps of the people from the background without artifacts. The rendered human must be planted in an artificial setting with a consistent lighting appearance.

The solution of these problems will make real time rendering of faces truly seamless.

Bibliography

- [ap96] E.H. Adelson and A.P. Pentland. *The perception of shading and reflectance*. Perception as Bayesian Inference, Cambridge University Press 1996, 409-423.
- [agam97a] Stefan Agamanolis and Alex Westner and V. Michael Bove Jr.. *Reflection of Presence: Toward More Natural and Responsive Telecollaboration*. Proceedings SPIE Multimedia Networks, 3228A, 1997.
- [agam97b] Stefan Agamanolis and V. Michael Bove Jr.. *Multi-Level Scripting for Responsive Multimedia*. IEEE MultiMedia, Volume 4, Number 4, October 1997.
- [alte92] T. D. Alter. *3D Pose from 3 Corresponding Points Under Weak-Perspective Projection*. MIT Artificial Intelligence Laboratory Memo, Number 1378, July 1998.
- [beck95] S.C. Becker and V.M. Bove Jr.. *Semi-automatic 3-D model extraction from uncalibrated 2-D views*. Proceedings SPIE Visual Data Exploration and Analysis II, 447-461, Feb, 1995, San Jose, California.
- [beym95] David Beymer. *Vectorizing Face Images by Interleaving Shape and Texture Computations*. MIT Artificial Intelligence Laboratory Memo, Number 1537, September 1995.
- [blac96] Michael J. Black and Allan D. Jepson. *EigenTracking: Robust matching and Tracking of Articulated Objects Using a View-Based Representation*. International Journal of Computer Vision, February 1996.

- [brad98] Gary R. Bradski. *Computer Vision Face Tracking For Use in a Perceptual User Interface*. Intel Technology Journal, Second Quarter, 1998.
- [breg97] Christoph Bregler, Michele Covell, and Malcolm Slaney. *Video Rewrite: Driving Visual Speech with Audio*. ACM SIGGRAPH, 1997.
- [brun95] Roberto Brunelli and Tomaso Poggio. *Template Matching: Matched Spatial Filters and Beyond*. MIT Artificial Intelligence Laboratory Memo, Number 1549, October 1995.
- [cogn98] Cognex Corporation. *World Wide Web*. www.cognex.com, 1998.
- [cybe98] Cyberware Laboratory Incorporated. *World Wide Web*. www.cyberware.com, 1998.
- [craw92] Ian Craw, David Tock, and Alan Bennett. *Finding Face Features*. Proc. European Conference on Computer Vision, May 1992.
- [dai96] Ying Dai and Yasuaki Nakano. *Face-Texture Model Based on SGLD and its Application in Face Detection in a Color Scene*. Pattern Recognition, 1996.
- [darr93] Trevor J. Darrell and Alex P. Pentland. *Recognition of Space-Time Gestures using a Distributed Representation*. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 1993.
- [deca98] Douglas DeCarlo, Dimitris Metaxas and Matthew Stone. *An Anthropometric Face Model using Variational Techniques*. SIGGRAPH, July 1998.
- [eise97] Peter Eisert and Bernd Girod. *Facial Expression Analysis for Model-Based Coding of Video Sequences*. Picture Coding Symposium, September 1997.
- [gard97] William G. Gardner. *3-D Audio Using Loudspeakers*. MIT Phd Thesis, September 1997.
- [hein98] Jochen Heinzmann and Alexander Zelinsky. *3-D Facial Pose and Gaze Point Estimation using a Robust Real-Time Tracking Paradigm*. Proceedings of International Conference on Automatic Face and Gesture Recognition, 1998.

- [horn86] Berthold Klaus Paul Horn. *Robot Vision*. MIT Press, 1986.
- [izqu97] Ebroul Izquierdo M.. *Stereo Matching for Enhanced Telepresence in Three-Dimensional Videocommunications*. IEEE Transactions on Circuits and Systems for Video Technology, Volume 7, Number 4, August 1997.
- [jeba97] Tony S. Jebara and Alex Pentland. *Parameterized Structure from Motion for 3D adaptive Feedback Tracking of Faces*. Appears in: Computer Vision and Pattern Recognition Conference, 1997.
- [krug96] Norbert Krüger, Michael Pöttsch, Thomas Maurer, and Michael Rinne. *Estimation of Face Position and Pose with Labeled Graphs*. Proceedings 7th British Machine Vision Conference, September 1996.
- [lam98] Kin-Man Lam and Hong Yan. *An Analytic-to-Holistic Approach for Face Recognition Based on a Single Frontal View*. Transactions on Pattern Analysis and Machine Intelligence, Volume 20, Number 7, July 1998.
- [lawr97] Steve Lawrence, C. Lee Giles, Ah Chung Tsoi, and Andrew D. Back. *Face Recognition: A Convolutional Neural Network Approach*. IEEE Transactions on Neural Networks, Special Issue on Neural Networks and Pattern Recognition, Volume 8, Number 1, Pages 98–113, 1997.
- [mach96] David Machin. *Real-Time Facial Motion Analysis for Virtual Teleconferencing*. Proceedings of the 2nd Int'l. Conference on Automatic Face and Gesture Recognition, 340-344, Oct, 1996, Killington, Vermont.
- [mcco97] Stephen McConnell. *World Wide Web Publication: Welcome to BT's 3D Talking Head*. <http://www.labs.bt.com/showcase/head/doc.htm>, British Telecommunications plc, 1997.
- [oliv97] Nuria Oliver, Alex P. Pentland, and François Bérard. *LAFTER: Lips and Face Real Time Tracker*. Computer Vision and Pattern Recognition, 1997.
- [omni98] Omniplanar Incorporated. *World Wide Web*. www.omniplanar.com, 1998.

- [orig98] Origin Instruments Corporation. *World Wide Web Publication*. www.orin.com, 1998.
- [pede96] Federico Pedersini and Stefano Tubaro. *Accurate 3-D Reconstruction from Trinocular Views through Integration of Improved Edge-Matching and Area-Matching Techniques*. VIII European Signal Processing Conference EUSPICO-96, September 1996.
- [polh98] Polhemus Incorporated. *World Wide Web Publication*. www.polhemus.com, 1998.
- [purc88] Dean G. Purcell and Alan L. Stewart. *The Face-Detection Effect: Configuration Enhances Detection*. Perception and Psychophysics, Volume 43, Number 4, Pages 355-366, 1988.
- [rowl98] Henry A. Rowley, Shumeet Baluja, and Takeo Kanade. *Neural Network-Based Face Detection*. Transactions on Pattern Analysis and Machine Intelligence, Volume 20, Number 1, January 1998.
- [seit96] Steven M. Seitz and Charles R. Dyer. *View Morphing*. SIGGRAPH, 1996.
- [shan88] K. Sam Shanmugan and A. M. Breipohl. *Random Signals Detection, Estimation and Data Analysis*. John Wiley and Sons, page 27, 1988.
- [shas96] Amnon Shashua. *Computer Vision: Image Based Rendering*. Lecture Notes, 1996.
- [shew96] Jonathan R. Shewchuk. *Triangle: Engineering a 2D Quality Mesh Generator and Delaunay Triangulator*. First Workshop on Applied Computational Geometry, Pages 124-133, May 1996.
- [shok98] Ali Shokoufandeh, Ivan Marsic, and Sven J. Dickinson. *view-Based Object Recognition Using Saliency Maps*. Journal of Image and Vision Computing, 1998.

- [sinh94] Pawan Sinha. *Pattern Motion Perception: Feature Tracking or Integration of Component Motions?*. MIT Artificial Intelligence Laboratory Memo, Number 1415, October 1994.
- [sobo96] Karin Sobottka and Ioannis Pitas. *Segmentation and Tracking of Faces in Color Images*. Second International Conference on Automatic Face and Gesture Recognition, October 1996.
- [sung94] Kah-Kay Sung and Tomaso Poggio. *Example-based Learning for View-based Human Face Detection*. MIT Artificial Intelligence Laboratory Memo, Number 1521, December 1994.
- [turk91] Matthew Turk and Alex Pentland. *Eigenfaces for Recognition*. Journal of Cognitive Neuroscience, Volume 3, Number 1, Pages 71-86, 1991.
- [vett96] Thomas Vetter. *Learning Novel Views to a Single Face Image*. Proceedings of Automatic Face and Gesture Recognition, 1996.
- [view98] Viewpoint DataLabs. *World Wide Web Publication*. www.viewpoint.com, 1998.
- [waib96] Alex Waibel and Jie Yang. *A Real-Time Face Tracker*. Workshop on Applications in Computer Vision, 1996.
- [yang94] Guangzheng Yang and Thomas S. Huang. *Human Face Detection in a Complex Background*. Pattern Recognition, 27:1, Page 53-63, 1994.
- [yow96] Kin Choong Yow and Roberto Cipolla. *Scale and Orientation Invariance in Human Face Detection*. Proceedings 7th British Machine Vision Conference, September 1996.
- [yuil92] Alan L. Yuille, Peter W. Hallinan, and David S. Cohen. *Feature Extraction from Faces Using Deformable Templates*. International Journal of Computer Vision, 8:2, 99-111, February 1992.
- [zhan97] Liang Zhang. *Estimation of the Mouth Features Using Deformable Templates*. IEEE International Conference on Image Processing, 1997.

[zhan96] Liang Zhang. *Estimation of Eye and Mouth Corner Point Positions in a Knowledge-Based Coding System*. Digital Compression Technologies and Systems for Video Communications, SPIE Volume 2952, October 1996.