

Manipulation of Video Eye Gaze and Head Orientation for Video Teleconferencing

C. Lawrence Zitnick

Jim Gemmell

Kentaro Toyama

June 16, 1999

Technical Report

MSR-TR-99-46

Microsoft Research

Microsoft Corporation

One Microsoft Way

Redmond, WA 98052

Abstract

Many desktop videoconferencing systems are ineffective due to deficiencies in gaze awareness and sense of spatial relationship. Gaze awareness and spatial relationships can be restored by software if heads and eyes can be tracked in video, and then graphically manipulated. We discuss graphics algorithms for manipulating eye gaze and head orientation. Our system takes video input annotated with head and eye information and outputs an adjusted video with appropriate gaze and head orientation. Initial results demonstrate the viability and usefulness of such a system.

1 Introduction

For over 50 years we have been hearing that video conferencing is about to become ubiquitous. The experience, however, is that most people question its usefulness [Chap72, Gale89]—we could easily write a primer on "How to fail at video-conferencing". It would include tips like:

1. Have long audio latencies and poor audio quality.
2. Make sure no-one else has compatible equipment.
3. Make it much harder to initiate than a typical phone call.
4. Eliminate gaze awareness and the sense of space you would have in a normal group setting.

This paper addresses the last problem: the lack of gaze awareness and sense of space found in most desktop videoconferencing systems.

In face-to-face communication, gaze awareness, and eye contact in particular, are extremely important [Arg88]. Gaze is a signal for turn-taking in conversation. It also expresses attributes such as attentiveness, confidence, and cooperativeness. People using increased eye contact get more help from others, can generate more learning as teachers, and have better success with job interviews.

Unfortunately, eye contact and gaze awareness are lost in most videoconferencing systems. Because a videoconferencing participant looks at the images on their monitor and not directly into the camera, he never appears to make eye contact with the viewer (See Figure 1 and Figure 2(a)). Video for each participant is in an individual window, placed arbitrarily on the screen, so they also never appear to look at other participants (See Figure 1). Without any gaze-awareness, the video loses much of its communication value and becomes uninteresting.

Several attempts have been made to create gaze awareness and spatialized teleconferences using specialized hardware. The Hydra system [Sel95] uses a small display/camera pair for each participant, placed far enough from the user that their gaze at the display is virtually indistinguishable from gazing at the camera. Others have used half-silvered mirrors or transparent screens [Oke94] with projectors to allow the camera to be placed directly behind the display.

In contrast, we attempt to create eye-contact using computer vision and graphics algorithms in software only. This paper describes our first attempts at placing video in a 3-dimensional space, with gaze and head orientation corrected. The ultimate goal of our project is to

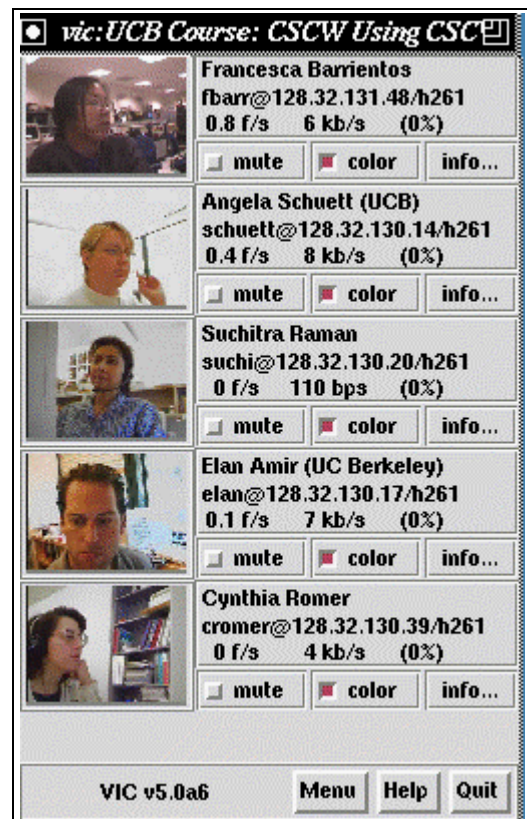


Figure 1: Videoconferencing: The typical videoconferencing interface does not provide gaze awareness or spatial relationships among participants

develop a spatialized videoconferencing application that supports a small (<5) number of participants from their desktops.

The system we envision consists of two components: a vision component for each user which tracks the user's head orientation, the approximate direction of their eye gaze, and the visible part of the user's eyeballs, and a graphics component which is capable of manipulating head orientation and eye gaze. This paper discusses the graphics component and associated issues only.

We consider two different techniques to correct eye gaze. The first is to mask the visible part of the eyeballs in the input image and then to replace them with synthesized eyeballs that appear to look in the desired direction (see Figure 4 (a) and (c)). This technique is discussed in Section 2. As Figures 2 and 5 show, however, editing just the eyes can change the facial expression, as well. The second technique addresses this problem by modifying the overall head orientation, Figure 6(c). Reorienting the head redirects the person's gaze, which in turn can create eye contact. We experimented with two head reorientation methods. The first algorithm, described in Section 3, involves warping the image to give the impression of a head rotation. The second, described in Section 4, projects the face image onto a 3D-head model, which can then be rotated.

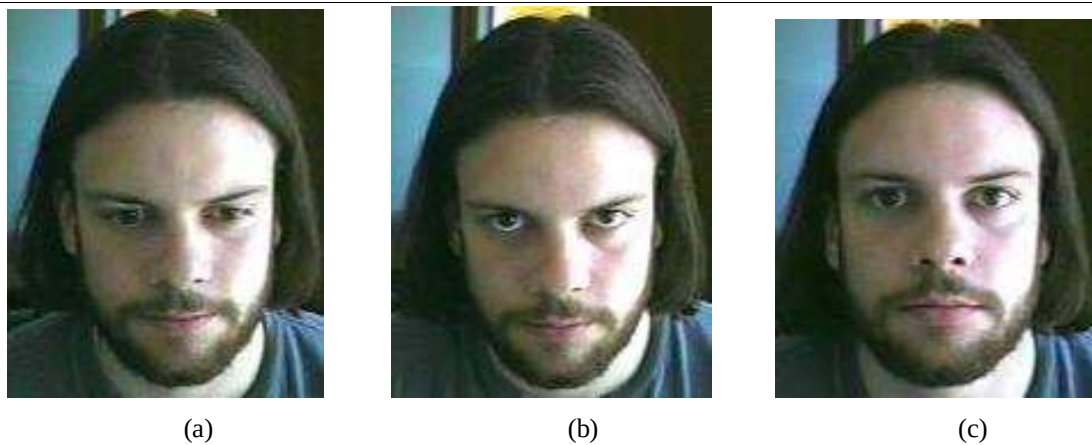


Figure 2: (a) Head and eyes directed away from viewer-- no gaze awareness and disinterested expression looking at something else; (b) Eyes directed at viewer, head directed away -- creates eye contact, but changes face to glowering expression; (c) Head and eyes directed at the viewer -- creates eye contact and correct facial expression of paying attention to you.

2 Eye Gaze Manipulation

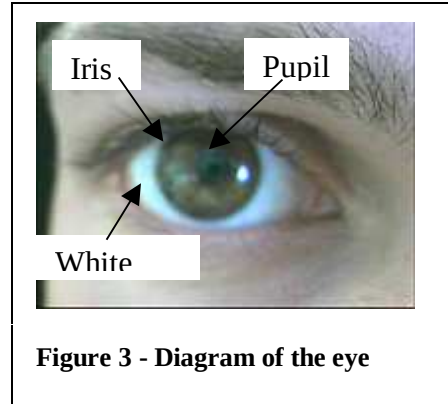
Assuming that the vision component has provided us with a segmentation of the eyes (the visible part of the eyeballs), we render new eyes, focused on the desired point in space, using computer graphics techniques. (Currently, we have successfully segmented the eyes in a few canned video sequences, which are used as input for our experiments. The vision problems here remain an area of future research.)

2.1 Drawing an Eyeball

The computer graphics task of drawing the eyes is relatively simple given the eye segmentation. We assume the average color of the white area, iris and pupil are known (See Figure 3). If we know the size of the eyeball the relative size of the iris can be estimated. We fix the radius of the pupil to be a fraction of the iris's radius. Dilation and contraction of the pupil are not currently modeled. To simplify rendering, the eyes are modeled without curvature. In practice, this is a reasonable approximation because the curvature of the eyes is not really noticeable until the head is significantly oriented away from the viewer (more than 30 degrees from our observations). Second, we assume that the object being viewed is at a constant distance so that spacing between the pupils (due to *eye vergence*) remains approximately constant (Section 2.3 relaxes this assumption).

The simplest technique for creating an eyeball is drawing two circles for the iris and pupil on a white canvas. Several additions could be made to give the eye a more realistic look:

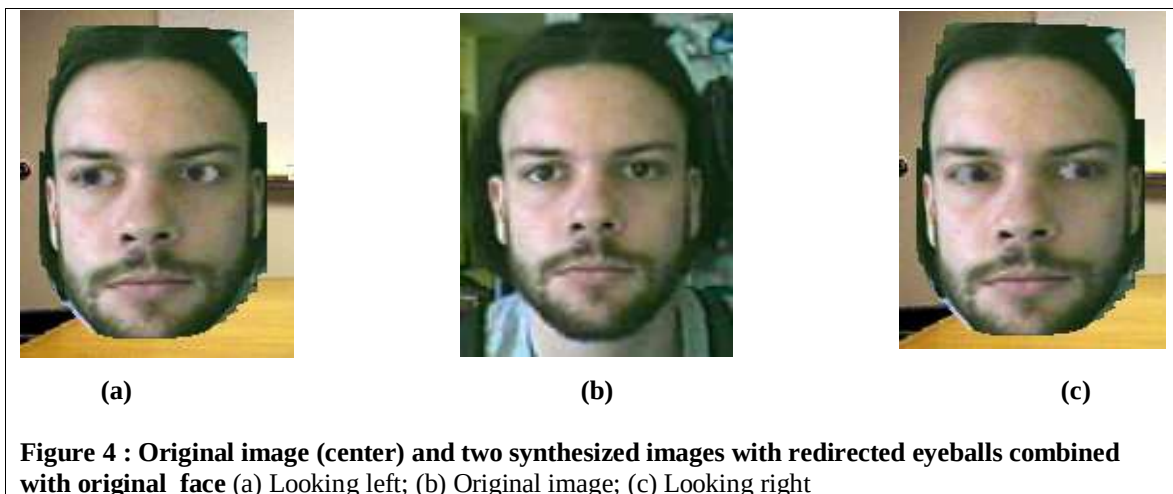
- A circle the color of the pupil, usually black, can be drawn around the edge of the iris. The iris's color typically becomes darker around the edges (the *limbus*).
- Random noise can be added to the iris and the white area to simulate texture in the eye. For smaller images this effect will not be noticeable and is not needed.
- The white area can be made darker towards the left and right edges of the eye. The white area is commonly darker towards the edges especially on the nose side.
- Highlights or spectral reflections can be drawn on the eyeball. As the iris and pupil change position the white spectral reflection should stay in a constant position relative to the general eye position, assuming head orientation doesn't change.



We have not yet implemented random noise and shading of the white area.

2.2 Drawing the Eye on the Face Image

Drawing an eyeball on a face requires two tasks. First, we draw the eyeballs on a temporary image. Second, we use the eye segmentation data to decide for each pixel whether to use the pixel information from the original face image or from the eyeballs image. The simplest method for combining the face image and eyeball image is to use color keying, similar to blue screening. Every pixel that is segmented as an eyeball pixel is colored blue (or whichever color is the color key color.) The eyeball image can then be blitted onto the face image. For a more refined or realistic look we can blend the edges of the eyeball with the face image using alpha values (See Figure 4).



2.3 Directing Eye Gaze at a 3D Point

Controlling eye gaze means controlling where the eyes are looking in 3D space. The 3D point that the eyes are looking at is called the *gaze point*. We want to determine the pupil positions that give the appearance that the eyes are focused on the desired gaze point. The eyes should converge or diverge as the gaze point moves closer or further away from the face. In order to compute the eye pupil positions we need to know (1) the 3D location of the eyeball centers, (2) the radius of the eyeball, and (3) if we have a 3D-head model we need to know its orientation and position in 3D space.

Recall that our model of the eyeball is flat. Thus, we only need to compute the plane on which to render the eye, and the center of the pupil. To make the computation easier, we make the approximation that this

plane is itself fixed with respect to the head. First, we rotate the head to be positioned upright looking down the Z-axis (the axis that is normal to the facial plane). The gaze point is rotated by the same rotation. For each eye, we then project the gaze point onto a Z-plane (a plane which holds Z constant – this is an approximation of the actual plane that should be used) using the eyeball center as the projection point. The Z-plane is placed in front of the eyeball center at a distance equal to the eyeball radius. We then intersect the line through the gaze point and the projection point with the Z-plane to find the projection of the gaze point on the Z plane. This point marks the center of the rendered pupil. Finally, the point is transformed by the inverse of the rotation first applied to the head.

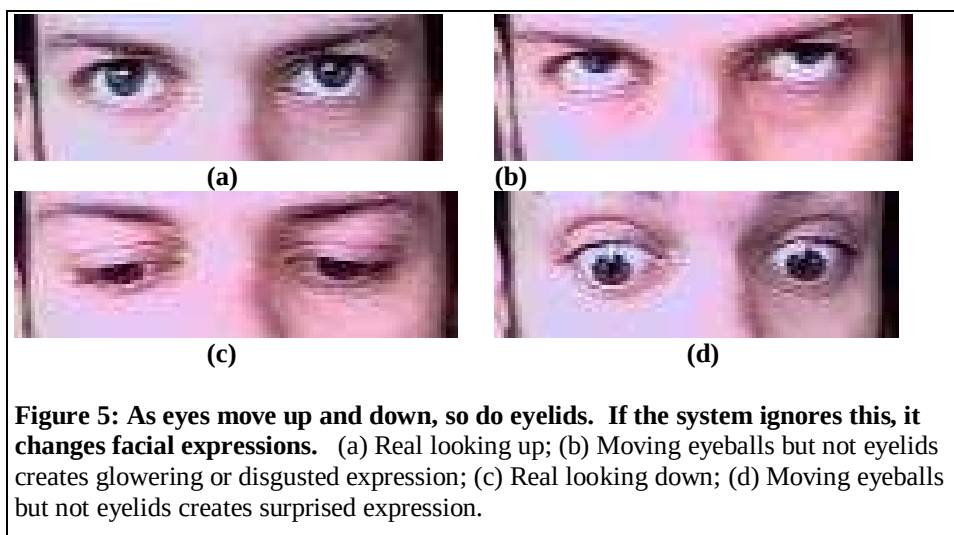
Is the use of the Z-plane model a reasonable approximation? When the head is oriented towards the viewer and the eye gaze is also directed towards the viewer, the pupil location is the same if the eyeball is modeled as a sphere or a plane. As the head rotates away from the viewer, the plane approximation becomes monotonically less accurate. This effect, however, is mitigated for three reasons: First, extreme combinations of head orientation and eye gaze (e.g., head facing left and eye gaze sharply to the right) are rare and difficult to modify for other reasons (tracking the eyes in such situations presents significant challenges for vision); thus, our project restricts eye gaze modification to instances when the head is oriented frontally, only (within ~30 degrees). Second, it is known that we are poor judges of eye gaze when a face is not oriented directly toward us [Arg88]; thus, the errors in approximation are unlikely to bother viewers.

2.4 Changing Eye Gaze and Facial Expression

When manipulating eye gaze as described above there can be an unfortunate side effect. The repositioning of the pupils can change the expression of the face. Typically when a person looks up or down the top eyelid follows the top of the pupil (See Figure 5(a) and (c)). Since the shape of the eye remains constant with our method the eyelid does not move up and down as it would naturally. As seen in Figure 5(b), when the top eyelid is too low it gives the face an expression of disgust. In Figure 5(d) the top eyelid is too high and the face appears surprised. There are two ways to change a person's eye gaze vertically without changing expression: the first is to synthesize the eyelids along with the eyeballs; the second is to warp the entire face appropriately (we will discuss the latter approach in the next section). Fortunately, the effect is minimal when the change in pupil position is small vertically. Changes in pupil position horizontally have little noticeable effect on facial expression.

3 Face Image Warping

So far we have only discussed moving the eyes. Now we move on to rotating the entire head to change gaze. This section describes our attempt at warping the face image using correspondence maps, and why we abandoned that approach in favor of a texture-mapped wire-frame approach.



Our first attempt to manipulate head orientation was based on warping the face image. Warping an image moves the pixels within an image. With a face image we would like to move the pixels to give the appearance of a face changing orientation. In order to get this effect we need to know where to move the pixels. The array that contains this information about how to move the pixels is called the correspondence map. Finding the correct correspondence map is the key issue in image warping.

For now let us assume that we would like to do a one dimensional image warp. An example using face images would be rotating a frontal view of a face to a right view of a face. The appropriate correspondence map would indicate how to move pixels in the frontal view in such a way that the right view results. There are some problems with occlusion and disocclusion, but only if the face is rotated enough to create significant self-occlusions.

So how do we find the correspondence map? The correspondence map can be automatically computed given two images, one of the frontal view and one of a right view. For each pixel in the frontal view, we can search within a 2D neighborhood around the same coordinate in the right view to find the corresponding pixel. To increase the likelihood of finding the best correspondence, we match small rectangular regions (templates) instead of isolated pixels. If camera calibration information is obtained for the two images the search area can be reduced to a 1D line [Sei96]. We will assume the camera calibration information is not known.

When searching within a 2D area around a pixel the size of the template can be varied to make tradeoffs between accuracy and precision. A lot of research with stereo vision has gone into finding the best window size. Many correspondence methods have also explored postprocessing the correspondence maps to remove noise. A review of stereo vision and optical flow research will yield many variations on the standard techniques [Ber87, Fua93, Kan94, Mar76, Hor81].

3.1 Image Warping Implementation

For our first task we took a frontal image of a face and warped it to be looking up. To find the proper warping we first computed the correspondence map between a frontal view of a face and a face looking up. Our first attempt at computing the correspondence maps used a standard stereo vision algorithm based on Sums of Absolute Differences [Oku93]. We did a search within a 2D area around each pixel in the frontal image with the image of the face oriented up. We tried many window sizes but the results always deformed the face, especially around the corners of the mouth and around the eyes. To compute a better correspondence map we used a method first proposed by [Fua93]. The method uses two initial correspondence maps to compute a final refined correspondence map. The two initial correspondence maps are the correspondences in each direction between the frontal image with the image of the face oriented up, i.e., correspondence from image A to image B and correspondence from image B to image A. Next, the matches which are consistent between the two correspondence maps are kept and the rest thrown out. To fill in the holes left by the inconsistent matches we interpolate between the consistent matches. This method improved the correspondence maps and the warping results. Warping the frontal face image to the position of the face in the other image looked realistic. Warping the image to rotate the face further up causes obvious deformations in the face. Using the inverse of the correspondence map to warp the face down worked well also as long as the face was not warped too far.

The next task was to combine two correspondence maps so we could warp the frontal face image left and up. We used the same techniques to compute each correspondence map. The correspondence maps were then linearly combined to compute the warping of the image. The results were good when warping either left or up. When the two correspondence maps are combined to move the face left and up, deformations are significant. For small angles (<5 degrees) it works fine, but has noticeable distortions with angles greater than 10 degrees.

3.2 Problems with Image Warping

After experimenting with image warping we decided not use it for several reasons:

- For every 5-10 degrees of rotation in every direction we would need a new correspondence map. This makes calibration difficult since many face images would be needed. The time to compute the correspondence maps could also be expensive.

- When combining correspondence maps to warp the face in a new direction, deformations are accumulated. Whenever large changes in orientation are needed the face could change its expression due to these deformations.
- Computing the correspondence between two faces is difficult. For template matching to work well we need Lambertian surfaces, i. e., surfaces which look the same from any angle, no spectral highlights. A face typically has many spectral highlights especially on the nose and forehead. To make matters worse much of the face, such as the cheeks and forehead, has no texture.
- When computing the correspondence maps we assume the face does not change shape. When the mouth moves up and down the shape of the face does change. Distortions will be noticeable around the chin area.
- When we use linear combinations of the correspondence maps we are changing the viewing position along a line in 3D space. Although there is no explicit rotation, we get the illusion that the face is rotating since there is only a small perceived change in rotation for each correspondence map. If larger angles were used when warping images with a single correspondence map the face would deform unnaturally. Seitz's view morphing using calibrated images could produce better results [Sei96].
- There is no easy way to compute the location of points on the face image relative to 3D points in the global world. This makes positioning the eye pupils to look at specific 3D points difficult.

4 Texture Mapping a 3D Head Model

Another method for changing the head orientation is texture mapping a 3D model. First a 3D model in the shape of the person's head is created. The face image is then projected onto the model in a process called texture mapping. After the face image is projected onto the model, the model can be rotated to any desired orientation (See Figure 6). This is an easier method to change head orientation than using image warping. The two methods are basically equivalent except the correspondence maps are in essence already computed with the generation of a head model.

4.1 Creating a 3D Head Model

When creating a 3D head model certain details are more important than others. When judging head orientation, two features are most important (from our empirical observations.)

1. First, the eyes must be modeled correctly. While the eyeballs themselves may be flat, the eye socket must be receded into the face. This is important to obtain a realistic look when rotating the head up and down.
2. Second, the nose must be modeled as protruding out of the face.

Other parts of the face which affect judgment of head orientation less – the the mouth, forehead, and cheeks – can be modeled by flat or slightly rounded surfaces. The model must be fitted separately for each user to account for differences in facial shape. Since the eye and nose are the most important features, the model is scaled to fit the face based on the geometric relationship of the eyes and nose. The amount the eyes are receded into the face and the amount the nose protrudes from the face are fixed. Once again, we assume the head will not be rotated more than 30 degrees, so the results should be realistic for a reasonable range of facial orientations.

4.2 Texture Mapping a Head Model

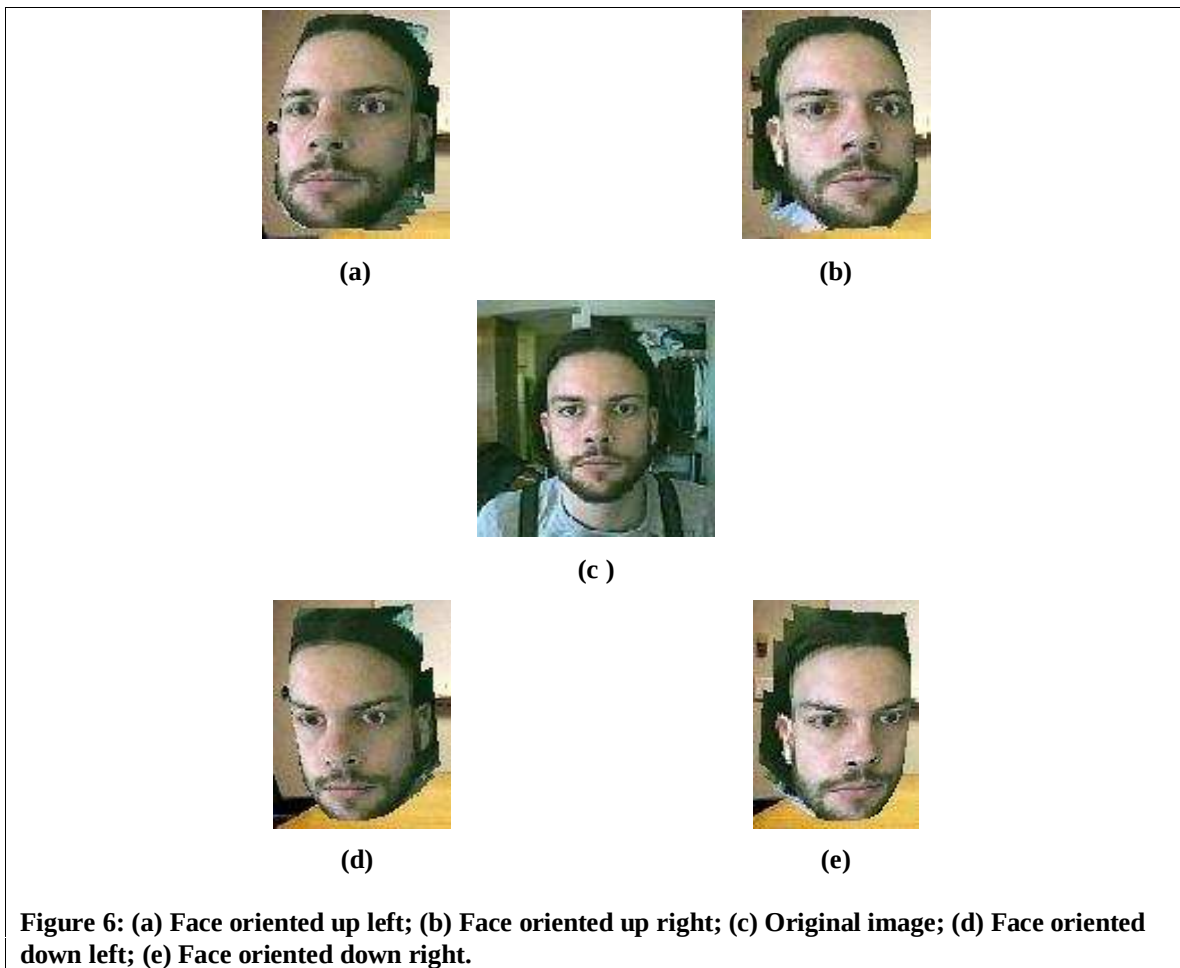
In order to texture map a model we need to know three values.

- The position of an anchor point on the 3D model and its location in the face image. We used the center between the nostrils. (This point is easy to track since it does not deform significantly when the face rotates or expression changes.)
- The orientation of the head in the face image. (This can be computed several ways. One method is to track multiple points on the face and compute head orientation from their relative spacing. Another method is to compute head orientation based on how a feature deforms, such as tracking the nostrils and computing head orientation based on how they change shape – see Appendix C.)

- The amount to scale the head model to correspond to pixel space. (This can be computed while head orientation is being computed.)

For each vertex of the head model we need to compute its 2D texture coordinates. The texture coordinates are the location in the face image that the vertex corresponds to. If we have the three values described above we can compute the texture coordinates for each vertex with the following steps. We will assume that we are rotating the model about the nose, however any point may be used.

1. For each vertex in the head model, subtract the value of the anchor point.
2. Rotate the head model to the same orientation as the head in the face image.
3. Scale the X and Y coordinates to correspond to pixel values. This is equivalent to doing an orthographic projection onto the image. An orthographic projection assumes all lines of sight are parallel, unlike the pinhole projection in which the lines of sight intersect at a point.
4. Add the 2D pixel location of the nose in the face image.



4.3 Dealing with Head Shape Changes

When a person is talking or changes expression the 3D shape of the head can change. The most obvious example is the chin moving up and down when a person is talking. If we assume our head model doesn't change shape this will cause problems. The easiest solution is to extend the wire-frame chin below its normal position. This way when the mouth is opened the chin texture will not slip down to the neck area.

When the mouth is closed the neck will appear to be attached to the bottom of the chin, but this will not be noticeable until the head is rotated significantly away from the viewer.

As stated before the eyes and nose are used most when judging head orientation. Fortunately the eye sockets do not change shape and the shape of the nose rarely deforms. Perhaps humans use the nose and eyes to judge head orientation because these features typically do not change shape. This is to the model's advantage, allowing us to use a static head model to achieve reasonable realism.

4.4 Inadvertent Changes of Expression

When rotating a head model away from the orientation of the head in the face image the features of the face can become deformed. Assuming the texture coordinates were found correctly, any deformations in the face are caused by the face model being incorrect. Many deformations go unnoticed, such as the side of the head being too narrow. Other deformations can actually cause changes of expression in the face. If the curvature of the mouth is incorrect in the head model, the mouth may appear to look either happy or sad when the head model is rotated up and down. The same may occur with the eyebrows, resulting in their appearing too tilted. Since the curvature of the mouth is different for everyone, finding an adequate solution to this problem may require "structure from motion" algorithms from vision, whereby the 3D structure of the face can be computed from a video sequence. Finding which parameters are important to ensure consistent expressions is by itself a difficult task.

If the orientation of the head within the face image is found incorrectly, the same effect can occur. Changes in expression will result when the face image is texture mapped onto the head model incorrectly. The problem will be most noticeable with errors in the vertical orientation of the head.

5 Conclusion

We have demonstrated two methods for manipulating gaze: eyeball synthesis and head reorientation.

We used a planar model for eyeballs. Eyeballs for low-resolution (128x128) face images can be realistically drawn given the correct segmentation.

We attempted two different algorithms for manipulating head orientation. The first involved image warping. For small angles of rotation (<5 degrees) we were successful in warping a person's head to a new orientation. For larger changes in rotation, realism was lost due to distortions. The correspondence maps that directed the warping of the face image are difficult to compute since the skin is a non-Lambertian surface. Errors in the correspondence maps resulted in distortions. Due to these problems and many more it was determined that the second algorithm for manipulating head orientation was preferable.

The second algorithm involved texture mapping a 3D-head model with a face image. The head model can then be rotated to the desired orientation. When the face image is successfully mapped onto the head model realistic rotations of 30 degrees or more can be obtained. Two problems still remain with this technique. First, the current head model is very crude. A more realistic head model and/or one that is adapted to different users could greatly increase realism. Second, computing the head orientation in the face image is inaccurate. When the orientation is tracked unsuccessfully, distortions are present in the final image seen by the viewer.

These early results in head and eye synthesis for gaze correction are encouraging. Vision systems still need to improve to meet the requirements of this synthesis. However, we are hopeful that such improvements can be made in the near term.

Gaze-corrected video takes information that was nearly useless from a communications point of view, and makes it extremely useful. We believe that gaze correction will be key to finally creating appreciation and demand for videoconferencing systems.

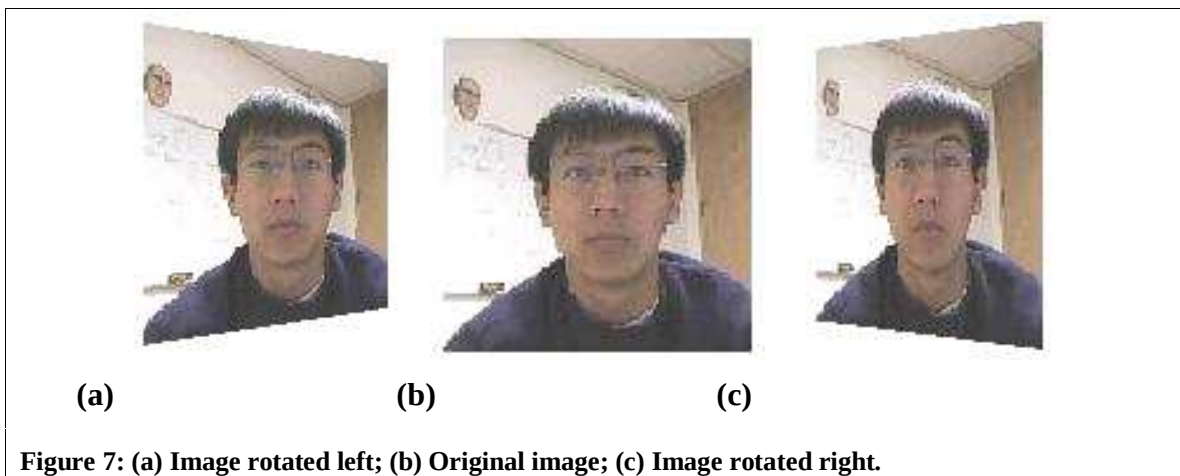
REFERENCES

- [Arg88] Michael Argyle, *Bodily Communication*, International Universities Press, Madison, Connecticut, 1988.
- [Ber87] Bernard, S.T. and Fischler, M.A., "Stereo Vision," in *Encyclopedia of Artificial Intelligence*. New York: John Wiley, 1987, pp. 1083-1090.

- [Chap72] Chapanis, A., Ochsman, R.N., Parrish, R.B., and Weeks, G.D. "Studies in interactive communication: II. The effects of four communication modes on the behaviour of teams during cooperative problem-solving," *Human Factors*, 14(6), 487-509, 1972.
- [Fua93] Fua, P., "A parallel stereo algorithm that produces dense depth maps and preserves image features," *Machine Vision and Applications*, vol. 6, pp. 35-49, 1993.
- [Gale89] Gale, S., "Adding audio and video to an office environment," *Proceedings of the 1st European Conference on CSCW*, London, 1989.
- [Hor81] B. K. P. Horn and B. G. Schunck, "Determining Optical Flow", *Artificial Intelligence*, 17, pp. 185-203, 1981.
- [Kan94] Kanade, T., and Okutomi, M., "A Stereo Matching Algorithm with an Adaptive Window: Theory and Experiment," *IEEE transactions on Pattern Analysis and Machine Intelligence*, 16(9), pp. 920-932, 1994.
- [Mar76] Marr, D., and Poggio, T., "Cooperative Computation of Stereo Disparity," *Science*, vol. 194, pp. 209-236, 1976.
- [Oke94] Okada, K.I., Maeda, F., Ichikawa, Y. and Matsushita, Y. "Multiparty videoconferencing at virtual social distance: MAJIC design," *Proceedings of CSCW '94*, pp. 385-395, 1994.
- [Oku93] Okutomi, M. and Kanade, T., "A Multiple-Baseline Stereo," *IEEE transactions on Pattern Analysis and Machine Intelligence*, 15(4), pp. 353-363, 1993.
- [Sei96] S. M. Seitz and C. R. Dyer, "View Morphing" *Proc. SIGGRAPH 96*, 1996, pp. 21-30.
- [Sel95] Sellen, A.J., "Remote conversations: The effects of mediating talk with technology," *Human-Computer Interaction*, 10(4), 401-444.

Appendix A: Mona Lisa Effect

The Mona Lisa effect occurs when a picture of a face appears to be looking at you even though the picture itself is tilted. This effect is named after the painting, whose subject appears to follow viewers with her gaze, no matter where they stand with respect to the painting (See Figure 7). There are two possible interpretations when looking at the Mona Lisa from an angle. First we could interpret the Mona Lisa as looking at a direction perpendicular to the picture frame. To make this conclusion we would need to observe that the Mona Lisa is looking forward and the picture is at an angle, therefore the Mona Lisa is looking in the same direction as the picture. The second interpretation is that the Mona Lisa is looking at us. It is known that humans are good at interpreting projective transformations. Therefore we can warp the picture inside our heads to be faced towards us; as a result the Mona Lisa appears to be looking at us. Due to this, attempting to change the eye gaze by rotating the image as if it were a plane will have no effect. Informal observation suggests that it might be possible to change the eye gaze by rotating the image if the face is not looking at the viewer. More works needs to be done to figure out the exact relationship between rotating an image and eye gaze.



Appendix B: Observations at SFMOMA (SF Museum of Modern Art)

Eyes – Eyes are very important except in pieces where the body's form was emphasized. The outline of the eye's white area was almost always present. The iris and pupil of the eye had more variation. A darker iris or an undefined pupil is present in paintings that expressed feelings of sadness or conformity. They also generally were looking down. In paintings that the eye gaze was especially important the iris was darker. A brightly colored iris with a clearly visible pupil typically expressed a happy, vibrant or sexy person. The eye gaze was typically pointed at the viewer. The use of color in the iris was used accordingly in a modern Chinese art exhibit. In all the paintings the irises were darker with typically no pupil. Most of these paintings expressed anger, conformity or sadness. Only one painting used brightly color irises; its subjects were smiling and looking directly at the viewer.

Nose – The nose played a secondary role in most pieces which emphasized facial emotion. Usually only a crude outline of the nose was present in the more abstract paintings. Some pieces left out the nose altogether, giving the impression that the subject was looking at the viewer. In pieces where the eye gaze is important more details are present on the nose. The tip of the nose and sometimes the nostrils were clearly visible.

Mouth – The mouth was only important in pieces expressing facial emotions. Either emotionless or crudely represented mouths were used with pieces emphasizing gaze.

One piece in particular demonstrated well which features of the face are used in gaze detection. A picture by Lucas Samaras was comprised of two photographs. One photograph was a frontal view of a man and the other was view of the same man with his head rotated slightly right. The artist cut out the eyes and nose of the photograph with the head oriented to the right and pasted it on top of the frontal view. When first viewing the piece you don't even notice that two photographs were used. The man's head definitely appears to be oriented to the right. If you put your finger over the eyes and nose, the head moves to a frontal view. The piece demonstrated well the importance of the eyes and nose, and lack of importance of the mouth and forehead in interpreting head orientation. A similar picture is shown in Figure 8.



Figure 8: Eyes/nose superimposed on face.

Appendix C: Nostril-Based Head Tracking

In order to compute the texture coordinates for the head model we need to know the position of a feature and the head orientation in the face image. An easy feature to track is the nostrils of the nose, especially when the camera is positioned below the user. We will assume the user's mustache isn't too large. Since the shape of the nostrils change as the head changes orientation, we can compute head orientation by tracking nostril deformations.

Positional Tracking

Tracking the position of the nose can be done using standard template matching techniques. If the user gives us the initial position of the nose, we can create a template of the nose. In each successive video frame we can search within a neighborhood of where the nose was located last to find the best match with the template. To reduce computation and to increase the size of the neighborhood searched, an initial coarse search can be done over a large area followed by a more refined search. To further increase speed, the resolution of the template matching can be reduced and only one band of the color space used.

Computing Head Orientation

When the head changes orientation the shape of the nostril will deform. If we can predict the deformation of the nostrils, head orientation can be computed. The first step is to create a set of nostril templates for a range of head orientations. These nostril orientation templates can be matched to the current face image to find head orientation. To create these templates an affine transformation of the tracking template is done. We will assume the nostrils lie on a plane with a known position and orientation. The plane is then rotated in 3D space and orthographically projected onto the camera's plane. The result is an affine transformation of the tracking template (See Figure 9).

