Very Low Frame-Rate Video Streaming For Face-to-Face Teleconference

Jue Wang[*], Michael F. Cohen[†]

[*] Department of Electrical Engineering, University of Washington

[†] Microsoft Research

Abstract

Providing the best possible face-to-face video over low bandwidth networks is a major challenge for current teleconferencing systems especially when implementing multi-party conversations. With the limitations of fixed webcams and low bandwidth, the transmitted faces may be significantly blurred, even unrecognizable. We describe a novel strategy for real-time coding of face video at a very low frame rate such as one frame per 2-3 seconds. By reducing the frame rate, we transmit high spatial fidelity face video at a very low bit-rate of 8Kb/s, with a reasonable representation of the original video, especially for conversants in "listening" mode. One challenge overcome is selecting which frames to transmit. We provide experimental results that show that our compression strategy provides very low bit-rate face-to-face teleconferencing.

1 Introduction

Face-to-face video communication is a potentially important component of real time communication systems. Inexpensive cameras connected to devices ranging from desktop machines to cell phones will enable video conferencing in a variety of modes such as one-to-one and multi-party conferences. Multi-party video conferences put an added strain on bandwidth requirements since multiple video streams need to be simultaneously transmitted. We concentrate in this paper on producing a very low bit rate result by trading frame rate to maintain visual quality. We hypothesize that this it is particularly appropriate for transmitting imagery from persons who are not currently speaking, although the same methods apply to all participants of a video conference.

Video teleconference solutions, such as PolycomTM [1], are specifically designed for broadband networks and can not be applied to low bandwidth networks. Different approaches have been proposed to reduce the bandwidth requirments, such as

[*]Address: Department of Electrical Engineering Box 352500, University of Washington, Seattle, WA 98195, Tel: +1-206-543-2150; Fax: +1-206-543-3842. juew@ee.washington.edu.

[†]Address: Microsoft Corporation, One Microsoft Way, Redmond, WA 98052, Tel: +1-425-703-0134; Fax: +1-425-936-7329. mcohen@microsoft.com.

the MPEG-4 face animation standard [2] and H. 26x [3]. By taking advantage of face models, the MPEG-4 face animation standard can achieve a high compression ratio by sending only face model parameters. However, it is difficult to make the synthesized face look natural and match the original video. H.26x waveform-based coding techniques are fully automatic and robust, but are not efficient for low bit-rate face video since their generality does not take advantage of any face models. These two types of techniques are combined together in a recently proposed low bit-rate face video streaming system [4], where prior knowledge about faces are incorporated into traditional waveform-based compression techniques to achieve better compression performance.

Previous face video compression techniques compress and transmit the entirety of every video frame. Thus, reducing the bandwidth will of necessity degrade the image in every frame. We argue that there is a minimum for the allocated bits for each frame below which conventional compression techniques cannot produce visually acceptable results.

We propose a low frame rate face video transmission strategy, which can, in real-time compress and transmit face video at very low bit-rates, such as 8Kb/s. The main contribution of our work is the novel compression strategy: instead of compressing and transmitting every frame at a very low quality, we reduce the video frame-rate and send only important regions of carefully selected frames.

We employ a real-time face tracking method at the encoder to provide a simple means to clip out the most relevant portion of the video. We further use a template matching-based "good face finder" to select only a few good frames from the face video as the transmission targets. A regular sampling of frames will often result in one in which the person has their eyes closed, or worse yet, half closed. We have all experienced this when taking photographs. In a multiple frame per second video stream it is fine to include blinks, but at only one frame every two seconds, such a frame is quite disturbing. The good frames are selected based on two criteria: (1) they should be visually good faces, (e.g., eyes open); and (2) they should carry as much information as possible (e.g., the face). The encoder then compresses and transmits only these good frame regions. At the decoder, the good frames are rendered by an image morphing method to create a normal frame-rate face video. The morphing between frames keeps the face "alive" even though no new information is being received between the temporally sparse frames.

Our system can be used individually in very low bandwidth networks, or as a complement to existing video conferencing systems. In a teleconference involving a group of people, each person's face will be captured and transmitted to others. Since there is generally only one speaker at a time, we can transmit the face of the speaker with higher frame-rate, high quality video while transmitting all the listeners using low frame-rate video to save overall network bandwidth.

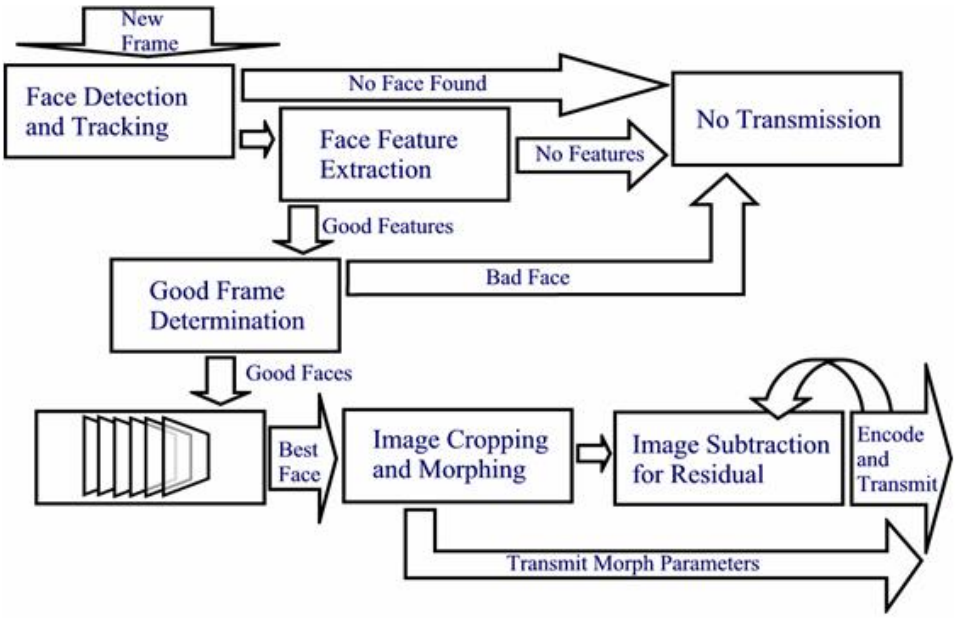

Figure 1: The system flow chart of the encoder.

2 The System

2.1 The Encoder

The flow chart of the encoder is shown in Figure 1. The input video frame is first processed to locate the face. We then track the positions of eyes, to be used both for face evaluation and motion compensation. The face and eye tracking, and face evaluation are described in detail in Section 3. We then evaluate the current frame based on two criteria and decide whether or not it is a good one. If it is, we perform a feature-based image morphing to align this frame to the previous transmitted frame. The current frame is subtracted from the previous frame and the residual is sent to a standard video encoder. We then transmit the compressed frame along with control parameters such as time stamps and face feature positions.

2.2 The Decoder

The flow chart of the decoder is shown in Figure 2. First we decode the received bits with the help of the previous decoded frame to recover the aligned image. We then use the transmitted control parameters to morph the face to its original location. We put the new frame into the frame buffer, and render the current displayed frame by image morphing between consecutive received good frames.

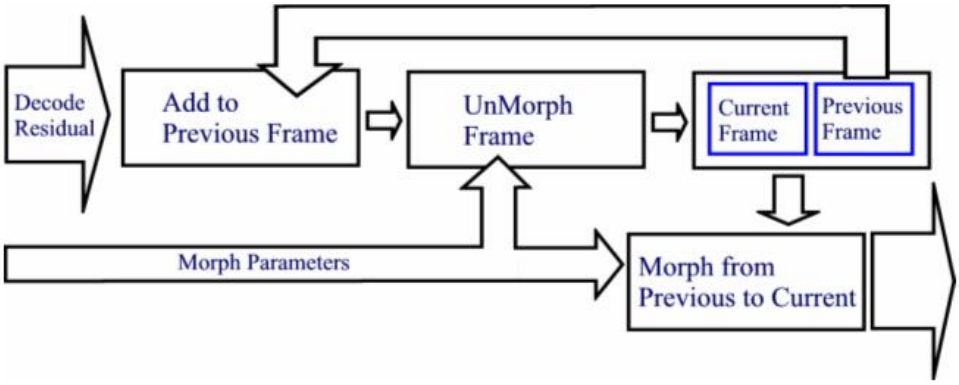

Figure 2: The system flow chart of the decoder.

3 Finding Good Faces

The low frame rate setting provides the freedom to choose which frame to transmit. For example, if the camera operates at 15 fps, but we wish to transmit only one frame every 2 seconds, then we have up to 30 frames to choose from (however, in practice we may wish to limit the choice to minimize latency). Since each frame will be seen for 2 seconds, it becomes critical to select "good" frames.

Choosing the specific features that distinguish a "good" frame from a "bad" one is somewhat objective. In an informal study the authors and colleagues examined many frames and judged their quality. We had originally hypothesized that some aspects of the eyes and mouth would correlate with the subjective judgment. In fact, only how open the eyes are had a significant correlation with "goodness." Examining Figure 3, one sees a typical set of frames from a video sequence. The openness of the eyes decreases from (a) to (b) to (c) to (d) and so did the subjective of opinion about the "goodness" of the frames. This is consistent with phycological research results [5]. No other aspect of the face had a consistent impact.

If we randomly select frames to transmit, we will frequently encounter blinking, or half-blinking eyes, as shown if Figure 3. To avoid such frames, we employ a face feature-based eye blink detection algorithm, which distinguishes between open and closed eyes.

3.1 Real-time Eye Tracking

Face tracking has been extensively used for efficient face video compression [6, 7]. We begin with the efficient face detection algorithm proposed in [8] to locate a rectangular box containing the face. For real-time detection and tracking, the detector scans only in the neighborhood of the face location in the previous frame.

Given the face box, we search the upper part of the box for eyes. Eyes from different people or under different illumination conditions may have significantly different appearances. For robustness, we employ a template matching based method for both eye tracking and blink detection. For each user, we manually indicate the pupil po-

Figure 3: Examples of good faces (a and b) and not so good faces (c and d). Considering the importance of eye contact in face-to-face communication, we only transmit faces with open eyes.

sitions on the first frame with wide open eyes. (We also ask the user to indicate the mouth corners for later morphing.) Two image patches are extracted at these positions as templates, one for each eye, as shown in Figure 4(a). On each frame, we iteratively match the templates to possible locations and select the best matching positions as the eye locations.

For efficiency and robustness to illumination changes, we use image feature based matching instead of directly comparing image patches. As shown in Figure 4(b), for an image patch, we compute its corresponding grayscale image, horizontal and vertical edge maps created with Sobel filters. By summing columns we further project the image patches to the horizontal axis (for the grayscale image and vertical edge map) or to vertical axis by summing rows (for the horizontal edge map) to produce three 1D signals. The similarity between two patches is computed as weighted sum of the correlations between corresponding 1D signals.

Mathematically, the three 1D signals for the left eye template T_L are denoted as $G_i^{T_L}, i = 1, ..., X_L$ (for grayscale image), $H_i^{T_L}, i = 1, ..., Y_L$ (for horizontal edge map) and $V_i^{T_L}, i = 1, ..., X_L$ (for vertical edge map), where X_L and Y_L are the width and height of the template. For a candidate image patch I, the three corresponding signals are denoted as G_i^I , H_i^I and V_i^I. The correlation between the two image patches is computed as

$$S(T_L, I) = w_G \cdot S\left(G^{T_L}, G^I\right) + w_H \cdot S\left(H^{T_L}, H^I\right) + w_V \cdot S\left(V^{T_L}, V^I\right) \qquad (1)$$

where w_G, w_H and w_V are predefined weights. We set them to be 0.4, 0.3 and 0.3, respectively. $S(A, B)$ is the signal correlation function computed as

$$S(A, B) = \sum_{i=1}^{L} (a_i b_i) \Big/ \sqrt{\sum_{i=1}^{L} a_i^2 \sum_{i=1}^{L} b_i^2} \qquad (2)$$

where L is the length of the signal.

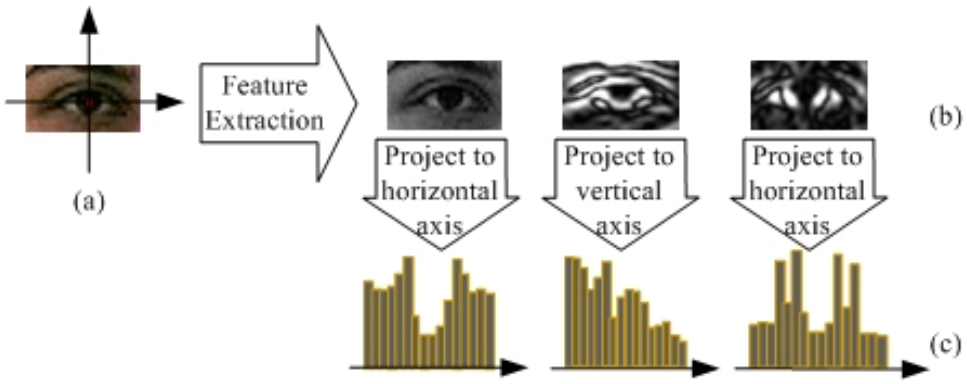

Figure 4: (a). Original left eye template. (b). Three feature patches computed from (a). From left to right: grayscale image, horizontal edge map and vertical edge map. (c). Three 1D features created by projecting the feature patches to horizontal or vertical axis.

3.2 Eye Blink Detection

The advantage of template-matching based eye tracking is that it not only gives us the best possible locations for the eyes, but also tells us how well the templates match to these locations, indicated by the computed correlation values. Since we use open eyes as templates, when the eyes are blinking, the correlation values drop significantly. Figure 5(a) shows the maximum correlation values on matching the left eye template to each frame of an example video. Blinks can be clearly seen at fairly regular intervals and they can be easily detected by applying a threshold on the correlation value computed from each frame. In our system we set the threshold to be 0.6.

Figure 5 shows some examples on face tracking and eye blink detection. The blue dots in Figure 5(b) indicate open eyes detected by the system, while the absence of blue dots in Figure 5(c) means that the system determines that the eyes are closed or half-closed.

Figure 5: (a). Maximum correlation values on matching the left eye template to each frame. (b) and (c). Face tracking and eye blink detection examples. Red boxes are face tracking box and green boxes are cropping windows. Blue dots in (b) indicate detected open eyes and the absence of blue dots in (c) indicates closed eyes.

3.3 Good Frame Selection

For each good frame F_G^i, there is a time stamp t_G^i. Good frames are selected from the original sequence based on the following criteria:

(1). $t_{min} \leq t_G^i - t_G^{i-1} \leq t_{max}$, where t_{min} and t_{max} are parameters determining how frequently we want to pick out good frames and essentially, they determine the required bandwidth of the transmitted video.

(2). Both the face tracker and eye blink detector give positive results, which ensures the selected face has good visual quality.

In cases that the user is temporally away from the camera, which means the second criterion cannot be satisfied, we force the system to send a random frame every t_{max} time to keep the transmitted video alive.

4 Compression and Rendering of Faces

4.1 Improved Motion Compensation

Selected good faces are compressed before transmission. The frames containing the good faces can be compressed using a standard video codec. Since the good faces are sparsely selected from the original video, the frame difference is typically larger than a high frame rate system making standard motion compensation less efficient. The face and eye tracking used to select frames can also inform the compression subsystem. First, by only transmitting the face region we avoid redundant transmission of the background. Second, the face tracking approximately aligns subsequent frames, significantly reducing the size of the interframe difference. Finally, by applying an image morph [9] that aligns the eye and mouth positions the difference between subsequent frames is further reduced.

We calculate the differences of eye locations between two adjacent frames, and estimate the location of the mouth corners based on the initial marked frame. From this we create a target morph mesh as in [9]. We morph each frame to its previous frame and subtract the result to obtain a residual image. In addition to coding the final residual we also transmit the eye and mouth locations so the process can be decoded.

4.2 Image Morphing for Rendering

The decoder (see Figure 2) performs the opposite operations. The residual is decoded and added to the previous frame. This new frame is then unmorphed based on the transmitted feature positions to produce the original frame.

We now have the option of simply switching from the old frame to the new one. To avoid a jarring jump, we can cross dissolve from the old to new but this produces ghostlike blurry images in the transition, as shown in Figure 6(e) and (f). We have found a better solution is to use the morph information to smoothly transform the old frame geometry to the new one while also performing the cross dissolve as was done in [9].

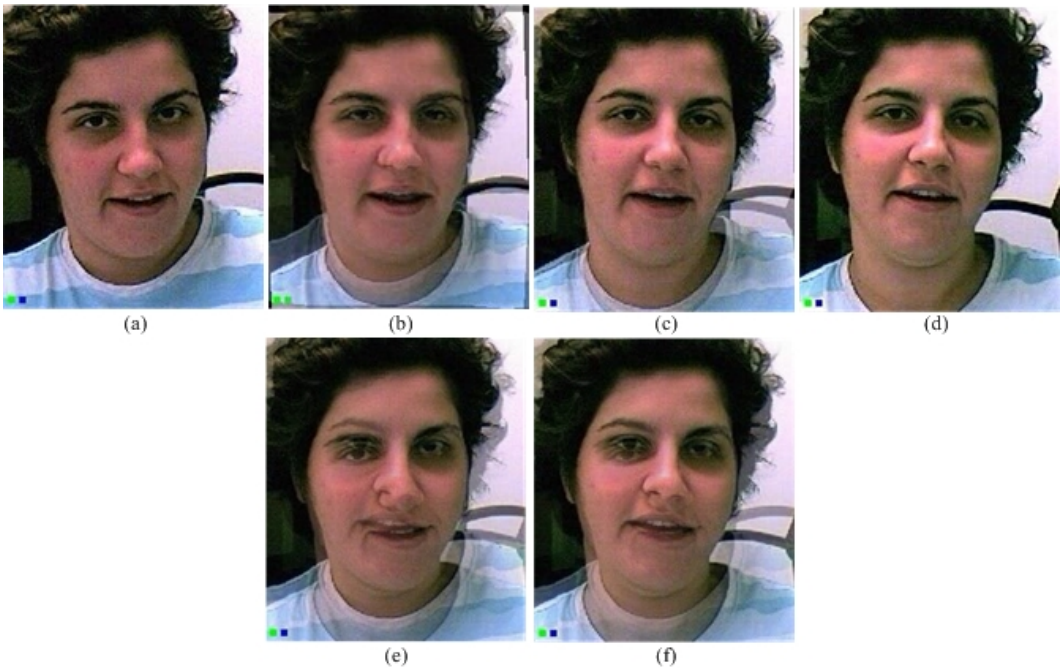

Figure 6: Examples of morphing-based rendering. (a) and (d). The first and the second decoded good faces. (b) and (c). Two synthesized intermediate faces created by image morphing and cross dissolving. (e) and (f). Two synthesized intermediate faces created only by cross dissolving.

In this way we can create a new video in the same frame-rate as the original video captured at the encoder. Figure 6 shows two decoded frames and the synthesized intermediate frames.

5 Results

5.1 Good Face Finding

As can be seen in Figure 7 we are able to select only good frames to transmit. The figure shows 5 of 30 frames selected for transmission with the good frame selection and by simply regularly sampling the video. The other 25 frames all looked good both with and without the selection criteria. It clearly shows that by using good frame selection, the transmitted video looks more natural, and facilitates important social-cognitive effects.

5.2 Compression

Table 1 shows the compression result of our system on a sample video with different settings of t_{min} and t_{max}, which control the desired frame rate. Note that our system only requires a very low bit-rate to transmit a semi-alive video. The last row of the table shows the compression result of our codec without using image morphing

Figure 7: Top: 5 out of 30 frames in the final video with good frame selection. Bottom: the corresponding set of frames in the final video without good frame selection.

for motion compensation. It clearly suggests that image morphing is very useful to achieve a better motion compensation in low frame-rate case.

Codec	Configurations	Bit-Rate
MPEG 2	640*480, 30f/s, good quality	322Kb/s
H.264	640*480, 30f/s, lowest quality	12.4Kb/2
Low-frame rate	240*280, $t_{min} = 1$, $t_{max} = 3$	7.4Kb/s
Low-frame rate	240*280, $t_{min} = 2$, $t_{max} = 4$	3.8Kb/s
Low-frame rate w/o morphing	240*280, $t_{min} = 2$, $t_{max} = 4$	5.4Kb/s

Table 1: Compression results of the low frame-rate system.

H.264 also achieves a low bit-rate compression, but the visual quality of the compressed video is significantly worse spatially, as shown in Figure 8. However, to be fair, it should be noted that the increased temporal frame rate does help depict a persons motion better. Further user studies will be needed to fairly examine the trade-offs. Informally, our system provides a better experience at a lower bit rate for viewing listeners, but may lose some important semantic meaning in the motion of speakers to maintain the better visual quality and lower bit rate.

More importantly, we perform the face/eye tracking and encoding in real-time as compared with our H.264 encoder which took about 30 minutes to encode a ten second video. Also, our system uses only about 1/3rd of the bandwidth of a slow dial-up line and thus is a good choice particularly when there are multiple participants in a video conference.

Figure 8: (a). One original frame. (b). Compressed by H.264 video codec (off-line) with a limited bit-rate at 12.4Kb/S. Note that the visual quality of face is poor. (c). Using low frame-rate compression system, we use only 1/3 bandwidth of dial-up connection to transmit high quality faces that look alive.

6 Conclusion

We have presented a real-time low frame-rate video compression system that allows the user to do face-to-face communication through extremely low bandwidth network. At the encoder side, the system is able to automatically select only a few good faces from the original sequence with high visual quality and compress and transmit them. At the decoder side, the system use image-morphing based rendering method to generate a normal frame-rate video. Experimental results show that the proposed system may be superior to more traditional video codecs for low bit-rate face-to-face communication.

References

[1] Polycom. http://www.polycom.com.

[2] ISO/IEC JTC1/SC29/WG11 N1902. Text for CD 14496-2 Video, November 1997.

[3] G.Côté, B. Erol, M. Gallant, F. Kossentini. 263+: Video coding at low bit rates. *IEEE Trans. Circuit and Systems for Video Technology*, vol. 8, no. 7, 1998.

[4] Z. Wen, Z.C. Liu, M. Cohen, J. Li, K. Zheng, T. Huang. Low Bit-rate Video Streaming for Face-to-face Teleconference. In *Proc. IEEE Int. Conf. Multimedia and Expo*, 2004.

[5] M. Garau, M. Slater, S. Bee, M.A. Sasse. The impact of eye gaze on communication using humanoid avatars. In *Proc. SIGCHI Conf. Human factors in computing systems*, 2001.

[6] W. Vieux, K. Schwerdt, J. Crowley. Face tracking and coding for video compression. In *Proc. of the First Int. Conf. on Computer Vision Systems*, 1999.

[7] J. Crowley, F. Berard. Multi-modal tracking of faces for video communication. In *Proc. IEEE Conf. Computer Vision and Patt. Recog*, 1997.

[8] S.Z. Li, X.L. Zou, et al. Real-time multi-view face detection, tracking, pose estimation, alignment, and recognition. In *IEEE CVPR Demo Summary*, 2001.

[9] T. Beier, S. Neely. Feature-Based Image Metamorphosis. *Computer Graphics*, 26(2), 1992.