Improving Depth Perception with Motion Parallax and Its Application in Teleconferencing

Cha Zhang [#1], Zhaozheng Yin [*2] and Dinei Florêncio [#1]

[#] *Communication and Collaboration Group, Microsoft Research*
One Microsoft Way, Redmond, WA 98052, USA
[1] {chazhang,dinei}@microsoft.com

[*] *Department of Computer Science and Engineering, Pennsylvania State University*
341 IST Building, University Park, PA 16802, USA
[2] zyin@cse.psu.edu

***Abstract*—Depth perception, or 3D perception, can add a lot to the feeling of immersiveness in many applications such as 3D TV, 3D teleconferencing, etc. Stereopsis and motion parallax are two of the most important cues for depth perception. Most of the 3D displays today rely on stereopsis to create 3D perception. In this paper, we propose to improve user's depth perception by tracking their motions and creating motion parallax for the rendered image, which can be done even with legacy displays. Two enabling technologies, face tracking and foreground/background segmentation, are discussed in detail. In particular, we propose an efficient and robust feature based face tracking algorithm that is capable of estimating the face's location and scale accurately. We also propose a novel foreground/background segmentation and matting algorithm with time-of-flight camera, which is robust to moving background, lighting variations, moving camera, etc. We demonstrate the application of the above technologies in teleconferencing on legacy displays to create pseudo-3D effects.**

I. Introduction

Three-dimensional contents such as games and 3D movies have regained a lot of interest recently. In August 2008, NVDIA introduced their GeForce Stereoscopic 3D driver technology, which allows people with compatible displays to play games in 3D with shutter glasses. Walt Disney announced in April 2008 that Disney and Pixar will make eight new 3D animated films in the next four years. DreamWorks Animation announced that it will release all its pictures in 3D starting 2009. With improving 3D capture and display technologies, it is foreseeable that 3D media will play more and more important roles in entertainment, medical applications, teleconferencing, arts, etc.

We humans perceive depth based on a number of factors: geometry, size, occlusion, focus, stereo disparity, motion parallax, etc. Most of the 3D displays today rely on stereopsis to trick our brains to think that the scene is 3D. While that is very effective, the rendered content do not change as the user moves around, hence the motion parallax is missing. In certain applications such as teleconferencing, motion parallax can be very important. For instance, in a multi-party teleconferencing, it is critical to make sure that the remote party's gaze can be faithfully duplicated, so that the meeting attendee can correctly sense the dynamics of the meeting. Without motion parallax, such gaze awareness is very difficult to achieve.

Another use of motion parallax in 3D is the creation of pseudo-3D effects on a legacy 2D display. Given the popularity of 2D display today and its excellent capability in rendering 2D documents, we expect there will be an extended period where legacy 2D display and 3D display (or 2D/3D convertible display, e.g., through masking, polarization or shutter glasses) will coexist. By introducing motion parallax, we will be able to give users with legacy 2D displays a sense of 3D (i.e., pseudo-3D), which could be an appealing effect.

In this paper, we constrain ourselves to the application of teleconferencing, and present two effective pseudo-3D effects, namely, *box framing* and *layered video*. In box framing, a virtual box is placed behind the display to create the illusion of viewing the remote party through a box, effectively placing the remote party *behind* the monitor. In layered video, the remote party's video is first segmented into foreground and background regions. The two regions are then rendered based on the user's head position, creating different occlusions between the two layers. The background can be further replaced by a different image for aesthetical or privacy considerations.

The main contributions of this paper, besides the application of pseudo-3D effects in teleconferencing, are novel algorithms for face tracking and foreground/background segmentation – two key technologies that enable the creation of such pseudo-3D effects. In particular, we develop a feature matching based tracking algorithm based on the efficient local image descriptor recently proposed in [1]. By estimating an affine transform between subsequent face regions using robust algorithms, we are able to compute the face location and scale very accurately, which is critical in order to control the 3D virtual viewpoint movement when performing the pseudo-3D rendering. For foreground/background segmentation, we propose to use a time-of-flight (TOF) camera to help improve the results. TOF cameras are active sensors that determine the depth value at each pixel by measuring the time taken by infrared light to travel to the object and back to the camera. These sensors

MMSP'09, October 5-7, 2009, Rio de Janeiro, Brazil.
978-1-4244-4464-9/09/$25.00

are currently available from companies such as 3DV Systems [2], Canesta [3] and Mesa Imaging [4] at commodity prices. Depth information measured by TOF is a reliable cue for segmentation, which is robust to lighting variations, camera movement, background object motion, etc. On the other hand, the depth image can be very noisy both spatially and temporally, which makes it infeasible to directly threshold the depth values to obtain a foreground mask. We propose a graph cut framework to combine the color and depth cues to achieve robust segmentation. A matting scheme that extends the closed-form matting method [5] is proposed to further improve the results around the segmentation boundary.

The rest of the paper is organized as follows. Two pseudo-3D teleconferencing effects for legacy 2D displays are described in Section II. The two technical components, face tracking and foreground/background segmentation and matting are presented in Section III and Section IV, respectively. Conclusions and future work are given in Section V.

II. Pseudo-3D Teleconferencing Effects for Legacy 2D Displays

The idea of introducing motion parallax to 2D/3D displays has been around for years, in particular in the virtual reality community. In such systems, head movement is usually tracked by dedicated motion sensors, such as magnetic sensor, optical sensor, radio and microwave sensor, etc [6]. While these sensors can provide very accurate localization results, they often require the user to wear additional equipment on the head, which is not user-friendly. In the work done by Pastoor et al. [7] and Suenaga et al. [8], stereo cameras were used to track the user's head and eye locations accurately. Their application is in the control of virtual worlds with head positions, where a 3D model of the virtual world is given. The recent work by Harrison and Hudson [9] further extended the idea to using a single webcam for tracking. They also considered the application of teleconferencing and adopted a layered representation of the video similar to ours, except that they tilt the layers instead of rendering the layers according the true motion parallax, mostly due to their lack of a reliable face tracker. In the following, we introduce two effective pseudo-3D effects for teleconferencing, namely, *box framing* and *layered video*.

1) Box Framing: Fig. 1 shows the basic idea of box framing. We create a virtual box, where the user shall see through to view the remote party. The remote party's video is pasted on a plane behind the virtual box, creating an interesting effect that the remote party is *behind* the display rather than *on* the display in regular video conferencing. Box framing does not require foreground/background segmentation, which is generally considered a very challenging task. An example of a conferencing session with box framing is available at:

http://research.microsoft.com/en-us/um/people/chazhang/boxframing.mpg

2) Layered Video: In layered video, the remote party's video is first segmented into foreground and background layers. These two layers are then set at different depths for 3D rendering, as shown in Fig. 2. An example layered video conferencing session is shown at:

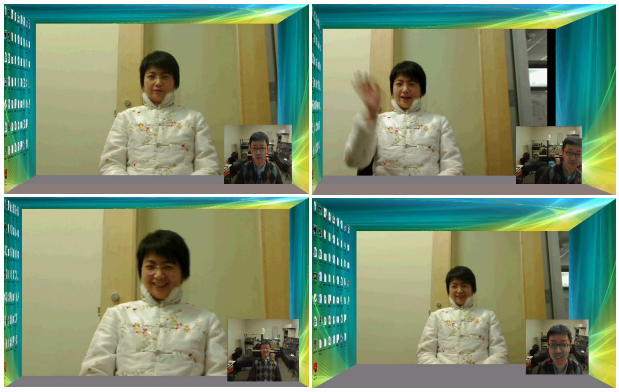

Fig. 1. Example frames for box framing. When the user moves around (as seen in the bottom right subwindow), the remote party's video is moved accordingly. The four box surfaces around the display boundary can improve the depth sense, and doubly used for extended data display.

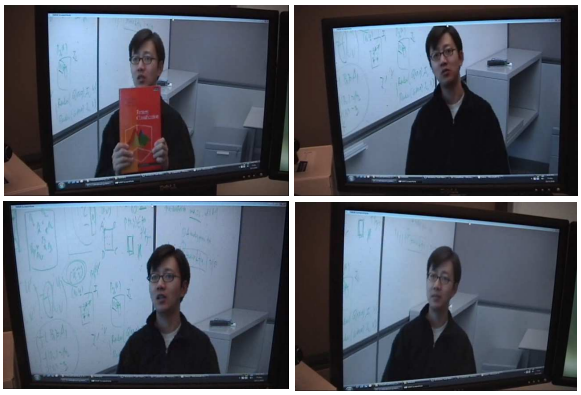

Fig. 2. Example frames for layered video. The remote party's video is first segmented into two layers. These two layers move accordingly based on the viewer's eye position.

http://research.microsoft.com/en-us/um/people/chazhang/layeredvideo.mpg

The challenge in creating the layered video effect is the layer segmentation. On this front, there have been a lot of work recently, including those using stereo cameras [10] and regular webcams [11], [12], [9]. However, these algorithms are often not very robust against moving backgrounds, lighting variations, camera movements, etc. In this paper, we experiment the use of time-of-flight cameras to help foreground/background segmentation, as will be detailed in Section IV.

III. Face Tracking

Tracking objects in videos has been a very important research topic in computer vision in the past few decades. Many successful algorithms have been developed and are suitable for faces, such as kernel based tracking [13], condensation based tracking [14], feature based tracking [15], etc. In this paper, we consider the special needs for tracking to create motion parallax, and propose a new feature-based tracking scheme based on some efficient local image descriptor recently

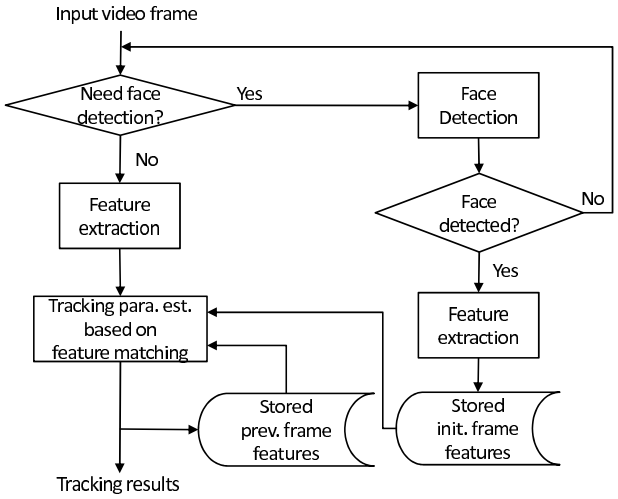

Fig. 3. The flowchart of the proposed face tracking algorithm.

developed in [1]. Note we assume only a single webcam is available for tracking, such that the algorithm can be broadly applicable to millions of users today.

To render a scene with motion parallax, the computer must estimate the current user's eye location in 3D. Since there is no depth information available with a regular webcam, one must rely on the scale of the face to compute the depth, which is very challenging for a typical face tracker. On the other hand, since the purpose of tracking is for creating motion parallax, it can be safely assumed that the face is mostly frontal during the process (the user shall be looking at the display), which is an easier problem than tracking faces with arbitrary poses.

Fig. 3 shows the basic flowchart of the proposed tracking algorithm. Given an input video frame, if no face has been detected, or the tracker has lost the tracked face, the face detector is launched to search for frontal faces in the scene. It then extracts features from the detected face and stores them for future feature matching. Given a new video frame, we also perform feature extraction around the previous tracking result, and compare these features with the ones extracted in the initial frame and the previous frame. An affine transform is estimated between the current frame and the previous frame, which is used to extract the 3D location of the user's face. Details of the tracking algorithm are presented below.

A. Feature Extraction and Matching

Feature matching based face tracking has been studied in the literature and is known for its accuracy in estimating the locations and scales of the faces, even for 3D tracking [15], [16]. The typical practice is to extract feature points in the previous frame, and then use Kanade-Lucas-Tomasi feature tracker [17] to locate and track features in the next frame. Alternatively, one can use independent feature extractors on both frames, and perform matching between the two feature sets using window-based comparison. If 3D tracking is performed, the window patches can be adjusted based on the face orientation, as was done in [16].

One issue with window based matching is that two features of the same point/patch may not match well if there is any significant change in orientation or scale, leading to errors in tracking. It has been shown that using multiple key frames can reduce errors caused by feature mismatching [16], but such a scheme will inevitably complicate the system development. In this paper, we adopt the feature extraction scheme and descriptor developed in [1], which is robust to changes in scale, rotation, illumination, noise, and changes in viewpoint. The feature points are first detected independently on the video frames with Harris-Laplace detector, and a 18-dimensional feature descriptor [1] is extracted for feature matching. Compared with other well-known feature descriptors such as the scale-invariant feature transform (SIFT) [18], the descriptor in [1] has a much lower dimension (18 versus 128 in SIFT), making it very suitable for feature matching in time-critical applications such as tracking, where the matching stage generally takes $30-40\%$ of the overall computation. We refer the reader to [1] for more details about the low-dimension embedded descriptor.

As shown in Fig. 3, the features are extracted on the current video frame, the previous video frame (stored), and the initial frame. After matching with descriptors, we obtain a set of corresponding feature pairs between the current video frame and the previous video frame, and a set of corresponding feature pairs between the current video frame and the initial video frame.

B. Tracking Parameter Estimation

Assuming that feature points on the frontal faces are roughly co-planar, we estimate an affine transform given a set of corresponding feature pairs. Let the feature pairs be $\{\mathbf{x}_{1i}, \mathbf{x}_{2i}\}, i = 1, \cdots, N$. We search for the best affine transform that satisfies:

$$\mathbf{x}_{2i} = \mathbf{A}\mathbf{x}_{1i} + \mathbf{b}, \tag{1}$$

where $\mathbf{A}$ (2×2 matrix) and $\mathbf{b}$ (2×1 vector) are the unknowns. This problem can be solved with minimum least squares. In order to avoid outlier feature pairs affecting the fitting quality, we adopt RANSAC for robust estimation. The number of matching point pairs after outlier removal is also an indicator of the confidence of the resultant affine transform. Once the affine transform is estimated, the location of the face can be obtained through $\mathbf{b}$, and the scale change of the face is determined by the *trace* of matrix $\mathbf{A}$.

As mentioned earlier, we have two sets of corresponding feature pairs, from which two affine transforms can be estimated – between the current frame and the previous frame, and between the current frame and the initial frame. The estimated scale and shift are then linearly weighted based on the number of matching feature pairs after outlier removal. Due to the excellent matching capability of the embedded feature descriptor, we found that it is sufficient to only use the initial frame as a mechanism to avoid and recover from drifting. The proposed tracker can reliably track faces within 45 degree rotation (in-plane and out-of-plane), which

is sufficient for pseudo-3D effects. The performance of the tracker can be seen in the box framing video in Section II-1.

IV. Foreground/Background Segmentation and Matting with a TOF Camera

The second technical component we will discuss is foreground/background segmentation and matting. While there have been a lot of efforts in live video foreground/background segmentation [10], [11], [12], most of them are very sensitive to environmental conditions such as lighting variations, moving background objects, moving cameras, etc. In this paper, we will use a Time-of-Flight (TOF) camera to help the segmentation task in these challenging situations.

A. Foreground/Background Segmentation

Foreground-background segmentation in images/videos can be considered as a binary labeling problem in a 2D/3D random field graph. Let $\{I_i\}$ and $\{s_i\}$ denote the sets of image pixels and corresponding labels respectively. Label $s_i = 1$ if I_i belongs to the foreground, and $s_i = 0$ otherwise. Let $\{V, E\}$ be a graph such that s is indexed by the vertices V and E contains all the pairwise links between two neighboring nodes (image pixels). Globally conditioned on the observation I, the joint distribution over labels s is

$$P(s|I) = \frac{1}{Z}\exp\left\{\sum_{i\in V}\sum_{k}\omega_k\Phi_k(s_i, I) + \sum_{(i,j)\in E}\Psi(s_i, s_j, I)\right\} = \frac{1}{Z}\exp\left\{E(s, I)\right\}, \quad (2)$$

where Z is a normalizing factor. $E(s, I)$ is the energy function to be minimized. The Φ_k's are region/pixel-based measurements (data association potentials) generated by different segmentation cues, and they are linearly combined with weights ω_k. Ψ represents pairwise interaction potentials between spatio-temporal neighboring nodes.

Different segmentation cues can be combined in a Conditional Random Field [19]. For example, stereo, color and contrast are fused in [10]. To avoid the inconvenience due to stereo camera calibration, motion information is explored in a monocular camera [11]. However, both the bilayer segmentation system [11] and background cut [12] are sensitive to background motion. In this paper, we explore the segmentation capability of a 3D image sensor (ZCam [2]) which provides real-time color and depth information, as shown in Fig. 4.

Data association potentials, Φ_k, measure how likely it is that node i has label s_i given image I without considering other nodes in the graph. The data energy term for color segmentation cue is defined as

$$\Phi_{color}(s_i, I) = -\ln P_{color}(I_i|s_i) \quad (3)$$

where $P_{color}(I_i|s_i = 1)$ represents the probability that I_i is a foreground pixel based on its color information. The foreground color likelihood is computed from the previous color image and foreground mask. Instead of modeling color likelihood using Gaussian Mixture Models ([11], [12]), we normalize the original color histogram as a non-parametric probability distribution which represents the actual color likelihood. A joint distribution in the RGB space (e.g. a smoothed color distribution in a $32bin \times 32bin \times 32bin$ RGB space) is used to represent the color likelihood. Fig. 5(a) shows the color energy term of Fig. 4. Similarly, we obtain $P_{color}(I_i|s_i = 0)$ from the previous color image and background mask. Note that other color spaces such as YUV and local color likelihood with spatial information can also be explored.

Fig. 4. Sample output from ZCam [2] (left: color image; right: depth image).

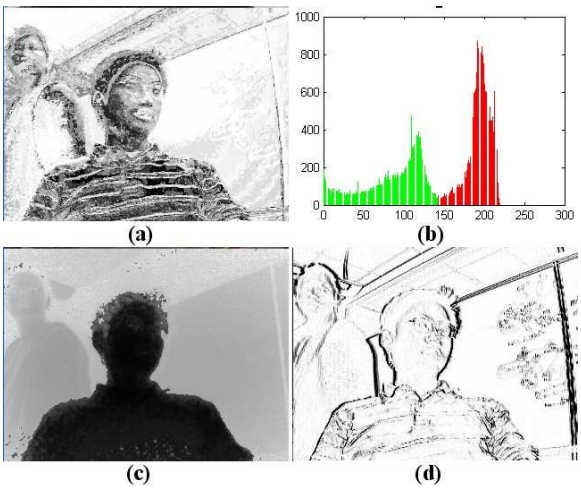

Fig. 5. Data and link terms in a graph. (a) $\Phi_{color}(s_i = 1, I)$; (b) depth histogram (the red and green parts belong to the foreground and background respectively); (c) $\Phi_{depth}(s_i = 1, I)$; (d) $\Psi(s_i, s_j, I)$.

To model the depth likelihood $P_{depth}(I_i|s_i)$, we run K-means algorithm on the 1D depth histogram to separate it into two clusters. As shown in Fig. 5(b), the depth map histogram consists of two clusters corresponding to foreground object and background respectively. After normalizing each cluster into a probability distribution, we define $\Phi_{depth}(s_i, I) = -\ln P_{depth}(I_i|s_i)$. Again, we use a non-parametric representation for depth likelihood. Fig. 5(c) shows the depth energy terms of Fig. 4.

The interaction potential between nodes s_i and s_j given image I is defined as

$$\Psi(s_i, s_j, I) = \frac{1}{dist(i, j)}\cdot e^{\frac{-\|I_i - I_j\|^2}{2\beta^2}}, s_i \neq s_j \quad (4)$$

where $dist(i, j)$ measures the Euclidian distance between nodes i and j; $\beta = <\| I_i - I_j \|^2>$ and $< \cdot >$ is the expectation operator. This potential describes how neighboring

nodes interact. For example, if the pixel color difference is small ($\| I_i - I_j \|^2 < \beta$), this potential encourages nodes s_i and s_j to have the same label. The temporal coherence between consecutive frames is computed in a similar way. Fig. 5(d) shows the link term between horizontal neighboring pixels.

The total energy $E(s, I)$ in Eq.2 is minimized by graph cut [20]. Our real-time foreground-background segmentation system is automatically initialized by the closest object to the camera at the beginning. Fig. 6 gives some segmentation samples where Fig. 6(c) shows the final minimized energy for each pixel. The foreground-background boundaries have high energy denoting where the cut happens in the graph.

After we achieve the binary foreground mask F and background mask B in the current frame, we update the weights of different segmentation cues for the next frame:

$$\omega_k = \frac{|m(P_k, F) - m(P_k, B|)}{\text{var}(P_k, F) + \text{var}(P_k, B)} \tag{5}$$

where $m(P, M)$ and $var(P, M)$ compute the mean and variance of the likelihood map P based on the mask M. The segmentation feature with high foreground-background discrimination power contributes more to the segmentation in the next frame, thus ω_k is dynamically adapted to scene changes.

B. Image/Video Matting

Using the ZCam with noisy depth maps, we build a live foreground-background segmentation system upon the graph-cut framework, but the hard binary segmentation lacks accuracy around the object boundary regions. Soft segmentation by image matting provides a solution to these problems (Fig. 6d). In this section, we discuss how to combine segmentation and matting techniques to improve the system performance.

Mathematically, image matting methods observe an image $\mathbf{I}$, which is assumed to be a convex combination of foreground image $\mathbf{F}$ and background image $\mathbf{B}$ through the alpha matte α,

$$I_i = \alpha_i F_i + (1 - \alpha_i) B_i + N_i, \tag{6}$$

where I_i, F_i and B_i stand for the intensity of the observed pixel, the foreground pixel and the background pixel at pixel location i; α_i can be any value in $[0, 1]$, which is the pixel's foreground opacity. N_i is the image sensor noise at each pixel, typically modeled as an i.i.d. zero-mean Gaussian random variable $N_i \sim \mathcal{N}(0, \sigma_I^2)$.

Based on this image model, closed-form matting [5] solves the alpha matte as

$$\alpha = \arg\min \left\{ \alpha^T L \alpha + \lambda (\alpha - b_s)^T D_s (\alpha - b_s) \right\} \tag{7}$$

where λ is some large number, D_s is a diagonal matrix whose diagonal elements are one for known pixels and zero for all other pixels, and b_s is the vector containing the specified alpha values for the known pixels and zero for all other pixel. L is a Laplacian matrix based on the local smoothing assumption.

Considering sparseness, color and gradient priors, we enhance the closed-form solution as

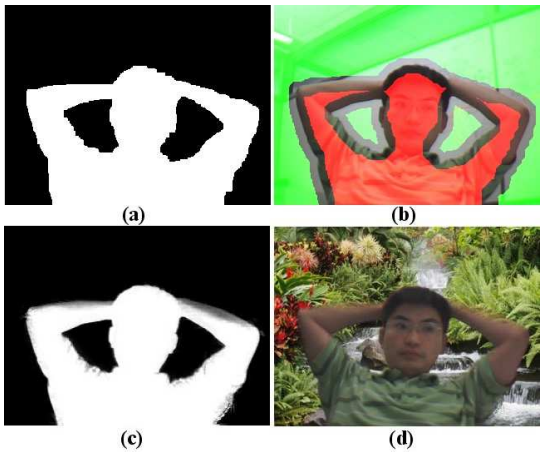

Fig. 7. Soft segmentation. (a) binary mask; (b) trimap: red pixels are known foreground $\alpha = 1$, green pixels are known background $\alpha = 0$, and others are unknown boundary pixels; (c) foreground matte; (d) foreground matted on a new background.

$$\begin{aligned} \alpha = \arg\min \alpha^T L \alpha + \lambda (\alpha - b_s)^T D_s (\alpha - b_s) \\ + \beta (\alpha - 1)^T D_C^f (\alpha - 1) + \beta (\alpha - 0)^T D_C^b (\alpha - 0) \\ + \gamma (\alpha - 0.5)^T D_G^L (\alpha - 0.5) \\ - \delta (\alpha - 0.5)^T D_G^S (\alpha - 0.5) \end{aligned} \tag{8}$$

where D_C^f is a diagonal matrix whose diagonal elements correspond to the foreground color likelihood for each pixel. D_C^b is for background color likelihood. The prior foreground and background probability distributions are computed globally from known color samples. D_G^L is a diagonal matrix whose diagonal elements are gradient magnitudes for pixels with large gradient and zero for pixels with small gradient. The intuition is that the alpha matte of the large gradient pixel around the boundary is between zero and one, or around 0.5. Since most of the alpha values in the matte are zero and one, we apply another sparseness prior to smooth image regions (i.e. pixels with small gradient) by minimizing $\alpha(1 - \alpha)$. Because $\alpha(1 - \alpha) = -(\alpha - 0.5)^2 + 0.25$, we can minimizing $-(\alpha - 0.5)^2$ equivalently. Hence, D_G^S is a diagonal matrix whose diagonal elements are one for pixels with small gradient magnitude and zero for other pixels.

The above objective function is quadratic, and we can obtain the matte by solving a sparse linear equation system. As shown in Fig. 7, we generate a trimap (Fig. 7b) by applying morphological operations on the binary foreground mask (Fig. 7a). The eroded binary mask (red) represents the known foreground and the complement of the dilated mask (green) represents the known background. The alpha matte for the unknown region is solved by the above enhanced closed-form matting method. Using the foreground matte (Fig. 7c), we can matte the foreground onto a new background (Fig. 7d).

We evaluate our enhanced closed-form matting method over a benchmark dataset [21] which includes eight images and ten different trimaps for each image. Table I compares our results

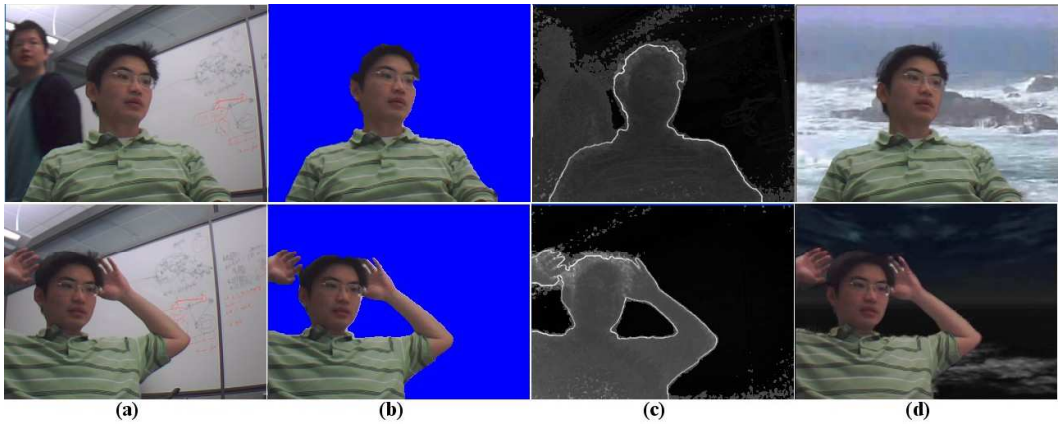

Fig. 6. Graph cut binary segmentation and soft segmentation by matting. (a) input color images; (b) graph cut binary segmentation; (c) final energy map of graph cut; (d) soft segmentation by matting.

TABLE I
EVALUATION ON THE BENCHMARK DATASETS (FORMAT: $error^{rank}_{minimum}$:$error^{rank}_{maximum}$). MINIMUM AND MAXIMUM ARE FROM TEN DIFFERENT TRIMAPS FOR A SPECIFIC IMAGE.

image	closed-form [5]	Robust [21]	Proposed
T1	79^3:97^3	46^1:85^2	47^2:67^1
T2	105^3:234^3	44^1:101^2	86^2:98^1
T3	61^3:80^3	38^1:67^1	45^2:74^2
T4	59^3:137^2	41^1:95^1	46^2:164^3
T5	23^3:356^3	10^1:155^2	12^2:104^1
T6	157^3:237^1	69^1:381^3	123^2:291^2
T7	77^3:143^2	31^1:165^3	71^2:111^1
T8	503^3:582^2	114^1:394^1	444^2:1117^3
Rank	3.0:2.4	1.0:1.9	2.0:1.8

with two state-of-the-art algorithms using the same parameters. We observed that adding the sparseness, color and gradient priors improves the performance over closed-from matting algorithm. The robust matting algorithm by [21] has slightly better results, but their methods are orders of magnitude slower than ours. An example video demonstrating the segmentation and matting on a challenging video is shown at:

http://research.microsoft.com/en-us/um/people/chazhang/LiveSeg.mpg

V. CONCLUSIONS AND FUTURE WORK

In this paper, we presented the idea of using motion tracking to introduce motion parallax for legacy 2D displays. Two pseudo-3D effects, box framing and layered video, were introduced for the application of teleconferencing. We then presented new schemes for two very important technical components – face tracking and foreground/background segmentation. The effectiveness of the proposed algorithms are demonstrated in a few videos available online.

REFERENCES

[1] G. Hua, M. Brown, and S. Winder, "Discriminant embedding for local image descriptors," in *ICCV*, 2007.
[2] "3dv systems," http://www.3dvsystems.com.
[3] "Canesta inc," http://www.canesta.com/.
[4] "Mesa imaging ag," http://www.mesa-imaging.ch/.
[5] A. Levin, D. Lischinski, and Y. Weiss, "A closed-form solution to natural image matting," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 30(2), pp. 228–242, 2008.
[6] G. Welch and E. Foxlin, "Motion tracking: No silver bullet, but a respectable arsenal," *IEEE Computer Graphics and Applications*, vol. 22, no. 6, pp. 24–38, Nov.-Dec. 2002.
[7] S. Pastoor, J. Liu, and S. Renault, "An experimental multimedia system allowing 3-d visualization and eye-controlled interaction without user-worn devices," *IEEE Trans. on Multimedia*, vol. 1, no. 1, pp. 41–52, March 1999.
[8] T. Suenaga, Y. Matsumoto, and T. Ogasawara, "3d display based on motion parallax using non-contact 3d measurement of head position," in *Proc. of the 17th Australia conference on Computer-Human Interaction*, 2005.
[9] C. Harrison and S. E. Hudson, "Pseudo-3d video conferencing with a generic webcam," in *Proceedings of the 10th IEEE International Symposium on Multimedia*, 2008.
[10] V. Kolmogorov, A. Criminisi, A. Blake, G. Cross, and C. Rother, "Bi-layer segmentation of binocular stereo video," in *Proc. IEEE Int'l Conf. on Computer Vision and Pattern Recognition (CVPR)*, vol. I, San Diego, CA, 2005.
[11] A. Criminisi, G. Cross, A. Blake, and V. Kolmogorov, "Bilayer segmentation of live video," in *Proc. IEEE Int'l Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2006.
[12] J. Sun, W. Zhang, X. Tang, and H. Y. Shum, "Background cut," in *Proc. Europ. Conf. on Computer Vision (ECCV)*, Graz, Austria, 2006.
[13] D. Comaniciu, V. Ramesh, and P. Meer, "Kernel-based object tracking," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 25, no. 3, pp. 564–577, 2003.
[14] M. Isard and A. Blake, "Condensation - conditional density propagation for visual tracking," *International Journal of Computer Vision*, vol. 29, no. 1, pp. 5–28, 1998.
[15] D. Decarlo and D. Metaxas, "Optical flow constraints on deformable models with applications to face tracking," *International Journal of Computer Vision*, vol. 38, no. 2, pp. 99–127, 2000.
[16] Q. Wang, W. Zhang, X. Tang, and H.-Y. Shum, "Real-time bayesian 3-d pose tracking," *IEEE Trans. on Circuits and Systems for Video Technology*, vol. 16, no. 12, pp. 1533–1541, 2006.
[17] C. Tomasi and T. Kanade, "Detection and tracking of point features," Carnegie Mellon University, Tech. Rep. CMU-CS-91-132, April 1991.
[18] D. Lowe, "Object recognition from local scale-invariant features," in *IEEE Int. Conf. on Computer Vision (ICCV)*, 1999.
[19] S. Kumar and M. Hebert, "Discriminative random fields," *International Journal of Computer Vision*, vol. 68(2), pp. 179–201, 2006.
[20] Y. Boykov and G. Funka-Lea, "Graph cuts and efficient n-d image segmentation," *International Journal of Computer Vision*, vol. 70(2), pp. 109–131, 2006.
[21] J. Wang and M. Cohen, "Optimized color sampling for robust matting," in *Proc. IEEE Int'l Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2007.