Virtual Mirror By Fusing Multiple RGB-D Cameras

Ju Shen, Sen-ching S. Cheung and Jian Zhao
Center for Visualization and Virtual Environments, University of Kentucky, United States, KY 40506
E-mail: {jushen.tom, sen-ching.cheung}@uky.edu, {Jian.Zhao}@microsoft.com

***Abstract*—Mirror is possibly the most common optical device in our everyday life. Rendering a virtual mirror using a joint camera-display system has a wide range of applications from cosmetics to medicine. Existing works focus primarily on simple modification of the mirror images of body parts and provide no or limited range of viewpoint dependent rendering. In this paper, we propose a framework for rendering mirror images from a virtual mirror based on 3D point clouds and color texture captured from a network of structured-light RGB-D cameras. We validate our models by comparing the results with a real mirror. Commodity structured-light cameras often have missing and erroneous depth data which directly affect the quality of the rendering. We address this problem via a novel probabilistic model that accurately separates foreground objects from background scene before correcting the erroneous depth data. We experimentally demonstrate that our depth correction algorithm outperforms other state-of-the-art techniques.**

I. Introduction

We are all familiar with the use of a mirror in daily life. There are many situations in which it is highly convenient if the mirror can depict virtual scenes alongside with the mirror image to elicit a desired visual feedback to the user. Imagine that we can try on new clothes or shoes without making a trip to the department stores. Same goes for hairstyling, cosmetics, and plastic surgery. The marriage between a mirror-like display and computer generated graphics presents the holy grail in such applications, and prototype systems have already been developed for these purposes [1, 2, 3].

The connection between manipulated mirror images and our cognitive functions, however, goes far beyond mere aesthetics. Perhaps the most well-known example is the use of the ”mirror box” in helping patients suffering from chronic neurological disorders such as phantom pain, hemiparesis from stroke, and other complex regional pain syndrome [4]. A mirror box is a simple top-open box with an opening on the side through which the patient can put her arm inside. There is a mirror in the middle facing the inserted arm and the mirror image creates an illusion of the presence of the opposite arm. It is this illusion that creates a visual feedback to the brain that alleviates the phantom pain caused by the amputated arm. While this form of mirror visual feedback is still not fully understood, it has already helped countless number of patients and has significant implication in our understanding of the elasticity of adult human brain in rewiring itself [5].

Equally mysterious is the special affinity towards self images among children on the autistic spectrum. Autism affects 1 out of 110 children and it is one of the fastest growing development disorders in the United States [6]. Autistic children have significant difficulty in socially interacting with others. One theory to explain such a poor social interaction is the indifference of human faces. Recent fMRI studies have shown low activity in the brains of autistic children compared with typically-developed peers when viewing pictures of human faces [7]. The same study, however, shows that these children maintain high level of brain activity when viewing pictures of themselves. While a full neurological explanation of self recognition is still under investigation, such affinity to self images can be capitalized in helping autistic children to learn from watching themselves, rather than others which they are likely to lose interest quickly. In fact, therapists have used edited self video to help children with autism modeling new behaviors. The therapist will record hours of video of the child, select appropriate footage and splice them together to depict the child seeming accomplishing tasks that are beyond his/her immediate capability. Such therapy is called Video Self Modeling or VSM and its effectiveness has been clinically demonstrated [8]. VSM involves significant amount of manual effort and cannot be used beyond teaching incremental improvements over existing skills. A ”programmable” mirror-like display can provide the flexibility in creating visual contents and the instant feedback to the patients that go far beyond the rigid video editing tools available today.

Simulating a mirror, however, is not simple. The naive setup of having a video camera on top of a monitor and showing the output of the camera on the monitor is clearly insufficient – the viewpoint is fixed for a camera while the mirror image depends on the position of the viewer. Thus, the challenge of simulating the mirror is to render different content on the display depending on the viewer’s perspective. To simulate a large mirror surface that can cope with wide displacement of a viewer, a camera-display system must be able to capture the 3-D world, track the moving viewpoint, render new view based on the viewpoint, and possibly add new visual content that are compatible with the scene geometry. In addition, it must be able to accomplish all these tasks in real-time otherwise it loses the instant visual feedback required to provide the realism of a mirror.

In this paper, we propose a camera-display system that simulates a mirror using a network of commodity RGB-D cameras in capturing 3D point clouds and color texture information. Our system uses the depth information to track the viewer, traces the light ray from each 3D point to the point of reflection on the virtual mirror, and determines the proper position and color values when the reflected ray hits the display surface. This model allows viewer-dependent rendering and supports arbitrary positioning or even movement of the virtual mirror.

Also this model fits naturally to a scalable client-and-server architecture which is essential in providing the required real-time performance. While the depth data provides important geometrical information of the scene, commodity RGB-D cameras typically suffer from significant noise and missing values problems [20]. As the quality of the depth data directly impacts the rendering, we propose a depth data denoising algorithm that models the noise process different between foreground and background, and applies different strategies in removing erroneous data and inpainting missing values. To the best of our knowledge, our proposed system is the first virtual mirror system that fuses multiple color and depth cameras in rendering realistic mirror image. All existing systems rely on a single camera or stereo camera pair in tracking viewpoints and rendering new views. Our system is capable of providing a far larger viewing surface and supporting a wider range of viewing position due to its capability in estimating the 3D coordinate for each pixel in the captured frame rather than merely getting the color intensity value on a 2D image as a normal camera does.

The rest of the paper is organized as follows: Section II reviews existing works in mirror simulation. Section III describes our proposed mirror model and provides details of implementation including a scalable client-server architecture to optimize the performance. Our depth denoising algorithm is described in Section IV. Experimental validation and performance measurement of our system can be found in Section V. Concluding remarks and discussion in future work are in Section VI.

II. Related Work

In the past, several research groups have developed virtual mirror prototypes. Though they differ in some aspects, most of them implement simple appearance modification with a limited viewpoint [9, 10, 11]. Darrell et al. described a virtual mirror interface that reacted to people by applying different graphical effect on their faces [9]. Similarly, in [10], the authors proposed a virtual facial modification program by user-driven 3D-aware 2D warping. However both of them did not consider the view point's influence on rendering virtual mirror. Franois and Kang designed a handheld mirror simulation device [12]. Although they considered the viewpoint change during the mirror image transformation, their system used a overly simplistic model by assuming the 3D world as a plane parallel to the mirror/imaging.

Virtual mirror has also been used in some very specific applications. In [3], a real time system is proposed for the real-time visualization of customized sports shoes. Similar systems have been developed to generate virtual mirror image for fashion [1, 2]. Since the target objects in these system are already known, they usually have a pre-computed model from either existing 3D model or collection of large training data. Therefore, the on-line computation can focus on rendering the correct texture on the generated image. In addition, the view point change is usually neglected in these systems due to its particular commercial intention.

One closely related subject of virtual mirror is view-dependent texture mapping (VDTM) [13, 14, 15]. Though our system shares many similar designs as in VDTM, it differs from the traditional VDTM in the following aspects. First, the geometry of the object in VDTM systems is usually known and represented as a pre-computed 3D mesh. In our system, we have a dynamic scene so the geometry needs to be estimated on-the-fly from the depth cameras. The incomplete description of the 3D world and the lack of object segmentation make generating a complete 3D mesh representation computationally impossible. Second, texture information in VDTM systems are usually recorded as planar image and image warping is a popular tool to map these texture into the 3D shapes. On the other hand, the network of image and depth camera pair in our system is able to obtain texture information at different viewing angles for every 3D point collected in the system. This provides a more realistic 2D rendering of the scenes. Finally, the viewpoint of our virtual mirror system is part of the 3D world captured and rendered by our system. Physical laws allow us to ignore certain aspects of rendering – for example, we cannot see the mirror when we are facing away from the mirror. No such restriction exists for VDTM.

A key component of our system is the use of commodity structured-light depth cameras such as Microsoft Kinect devices. Depth images obtained by such devices have distorted and missing depth values. Some of the missing depth values are caused by the disparity between the infrared (IR) light projector and the IR cameras. Others can be caused by absorption, poor reflection or even shadow reflection of the structured-light patterns. The depth noise problem is especially pronounced near depth discontinuities. A number of algorithms have been developed to denoise depth image using super-resolution [19, 21] and image in-painting [22]. A common theme among these work is to rely on information obtained from the companion color images to predict missing depth information. This strategy does not always work as color edges and depth edges do not necessarily coincide with other. In [19], Garro et al. presented an interpolation scheme for depth super-resolution. A high resolution RGB camera is used to guide the up-sampling process on the depth image. To interpolate the missing depth pixel, the scheme uses neighboring depth pixels mapped into the same color segment as the target pixel. This method relies strongly on the extrinsic alignment between the color and depth image. However, noisy depth values along object boundaries may map into a wrong color segment and propagate its effect to other pixels in the segment. In [21], a low resolution depth image was iteratively refined through the use of a high resolution color image. Bilateral filter was applied to a cost function based on depth probabilities. A final high resolution image was produced by a winner-takes-all approach on the cost function. These approaches work well for the super-resolution problem where missing depth pixels are uniformly distributed. Depth images obtained by structured-light sensors often have large contiguous regions of missing depth measurements which cannot be handled by such approaches. In [22], Wang et al. proposed a stereoscopic

in-painting algorithm to jointly complete missing texture and depth by using two pairs of RGB and depth cameras. Regions occluded by foreground were completed by minimizing an energy function. The system was cumbersome as an additional pair of color and depth cameras were needed.

III. Virtual Mirror Modeling and Implementation

To simulate the mirror experience, our model consists of three components:

1) Structure of the 3D scene.
2) 3D location of the viewpoint or more precisely, the optical center of the eye.
3) 3D location and pose of the mirror.

The basic work-flow of our system is illustrated in Figure 1. The system first collects and enhances the geometry and color data for individual cameras. It then records the scene information as a 3D point cloud and estimates the viewpoint position V. Next, it traverses each point S_i in the cloud to find the corresponding reflection point R_i on the mirror, which determines its reflection ray to hit the view point V. Once the reflection point is obtained, it computes the intersection point P_i between the display surface Π_d and $\overrightarrow{R_iV}$ on the display. A Z-buffer is used to determine if P_i is not occluded and indeed visible to the viewer. If so, the local coordinates of P_i on the display $[P_i]_d = (x_i, y_i)$ are calculated and the corresponding pixel value $I(x_i, y_i)$ is determined based on the color information stored at S_i and the viewpoint V. We defer the denoising process to Section IV. In the following subsections, we describe the rest of our proposed system in details and the speedup strategy used in conjunction with a client-server architecture.

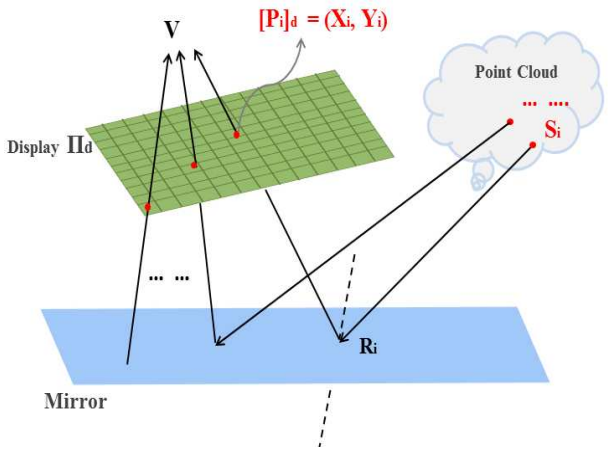


Fig. 1. Virtual Mirror Modeling

A. Point Cloud Generation

The 3D point cloud and the color texture information are obtained through a network of multiple fully-calibrated RGB-D cameras. As the image captured by the depth camera has the depth value available for each pixel. Based on the intrinsic parameters of the camera and the measured depth, we can obtain the corresponding 3D point by applying an inverse camera projection operation. Furthermore, for each RGB and depth camera pair, pixel-wise alignment can be achieved between the RGB and depth image according to their extrinsic parameters including the rotation and the translation. Thus, for each pixel on the color image, we can compute the tuple $\{X, Y, Z, R, G, B\}$ where R, G, B are the three color channels.

As there are multiple such camera pairs on the network, we need to unify the 3D point clouds generated by each of them. We first choose one color camera as the world coordinate system. We apply the same calibration procedure between this color camera with all the other color cameras and obtain corresponding extrinsic parameters as R_{1i} and T_{1i}, where $i = 2, 3, ...$ is the index of each depth-color camera pairs. Applying the transformation on all the point clouds except the one from the first camera, we obtain:

$$\begin{pmatrix} X_i' \\ Y_i' \\ Z_i' \end{pmatrix} = R_{1i} \cdot \begin{pmatrix} X_i \\ Y_i \\ Z_i \end{pmatrix} + (-R_{1i}T_{1i})$$

where $(X_i, Y_i, Z_i)^T$ represents a 3D scene point computed from the i^{th} color camera. $(X_i', Y_i', Z_i')^T$ represents the converted coordinate in the global camera's coordinate system.

B. Viewpoint Tracking

To provide viewpoint dependent viewing, the system needs to track the viewer's eyes' position. As our display only renders a monocular view, we track the head position rather than the actual locations of the two eyes. We approximate the head as a sphere and treat the center as our target viewpoint. While there are many high-performance sophisticated tracking algorithms in the literature, we take advantage of a single-user nature of the system and develop a very simple depth-based tracking algorithm. We assume that the user is much closer to the depth camera than the rest of the 3D scene points. As such, the histogram of depth values has a sharp peak of small values corresponding to the viewer that can be easily separated from the rest using a single threshold. These small depth values are then back-projected to the corresponding 2-D camera spatial coordinates. The actual depth values are not used anymore and we only keep the 2-D binary shape of the viewer. Morphological opening and closing are applied to fill in small holes and to smooth the outline of the silhouette. Starting from the topmost point of the silhouette, our algorithm follows the outline in both directions and calculates the curvature at each boundary point. The characteristic omega shape of a head induces a curvature curve that has a sharp dip from positive to negative at the two inflexion points. Detection of these two inflexion points define the extent of the head curve which are then used to fit a circle. The estimated center of the circle on the camera plane is temporally smoothed with a Kalman filter. The actual 3D coordinates of the viewpoint is then estimated to be the 3D point that minimizes the sum of distances to each of the line formed between each camera's optical center and the center of the corresponding head circle.

C. Virtual Mirror Rendering

After obtaining the 3D point cloud and the viewpoint, the next step is to render the mirror image from an arbitrarily-positioned virtual plane mirror onto the display surface. For each 3D scene point, the rendering step is equivalent to first identifying the reflection point on the mirror and then the display point on the display surface. The identification of the reflection and display points can be viewed as two consecutive procedures of virtual camera projection. As illustrated in Figure 2, the reflection point can be viewed as the projection of the scene point onto a virtual camera, denoted as "Virtual Camera 1" in the figure, with the image plane at the mirror and the optical center at the mirror image V' of the viewpoint V. The display point is the projection of the reflection point onto another virtual camera, denoted as "Virtual Camera 2" in the figure, with the image plane at the display plane and the optical center at V.

Thus, the rendering problem boils down to estimating the camera projection matrix for each of the two virtual cameras. The approaches to estimate the camera matrix for both cameras are identical and we use the display surface camera as an example. The focal length f is simply the distance between V and the display surface. The image center (c_x, c_y) on the display plane is the perpendicular projection V_p of V on the display plane relative to the top-left corner G_1. As for the extrinsics, the rotation matrix R can be inferred by the orientation of the display plane. For example, the plane normal is computed as $\overrightarrow{n} = \overrightarrow{G_1G_2} \times \overrightarrow{G_1G_2}$. R is determined according to the angle subtended by $\overrightarrow{n}$ and the Z axis in the world coordinate system. The translation T can be computed directly based on the distance between V and the origin of the world coordinate system.

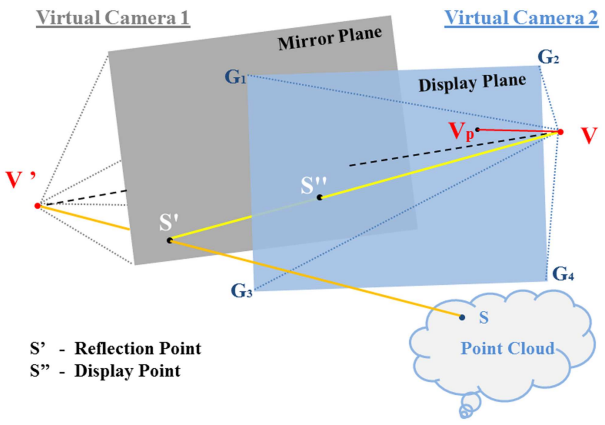


Fig. 2. Modified camera projection models for display rendering

Once all the above parameters are computed, the corresponding image on the display can be rendered based on the display points, denoted by S''. For each pixel on the display surface, there might be zero to multiple corresponding display points. If there is only one display point, that pixel will assume the color value associated with the corresponding 3D scene point S. If there are more than one display points, they can either be from the same 3D scene point but originated from different cameras, or they can be different scene points that fall on the same 3D ray VS''. For the first case, their depth values would be close to each other. For the second class, their depth values would be far apart and the one with the closest to the mirror would occlude the rest. This suggests a simple procedure of first clustering all the scene points that share similar depth values and then selecting the group that is closest to the mirror. To compute the final color among all points in the winning cluster, we use the scheme of *winner takes all* to select the one that best aligns with the viewpoint [9].

D. Scalable client-server architecture

We adopt a Server-Client distributed system to deal with larger rendering space and high computation complexity. Each client is responsible for one RGB-D camera. It also contains all the calibration information and the user's viewpoint from previous instance. Each client first computes the 3D point cloud in the world coordinate system and an initial estimate of the viewpoint. Based on the viewpoint from the previous instance, the client will render the mirror image, possibly incomplete, based on its own point cloud data. As such, the 3D point cloud processing, the viewpoint estimate and the mirror image rendering are all done at the client level. The mirror image and the viewpoint estimate are then sent to the server. The server then combines all the mirror images with the clustering algorithm to fill in any area that may be occluded from one camera but visible at the others. The viewpoint estimate is also refined by taking a statistical average of all estimates and broadcast back to all the clients for the rendering of the next frame.

IV. Depth Data Denoising

Noise present in the depth image can significantly impair the quality of the rendered mirror image. Erroneous depth can move background scene to foreground or vice versa. Missing depth values and occlusion can lead to "holes" in the rendered image. To handle these problems, we propose a novel depth data denoising algorithm based on foreground-background separation. By separating pixels into foreground and background, the generated image can be completed in a guided manner. Missing depth pixels are interpolated or inpainted by neighboring depth pixels from the same group to prevent smearing of object boundaries. In addition, the foreground-background separation admits a low-complexity and better-quality rendering of the mirror image – foreground pixels can be first extracted and projected onto the display image while the remaining regions are filled by pre-captured backgrounds, thereby avoiding the rendering of the entire frame and filling background holes caused by the change of viewpoint.

The foreground-Background separation is performed in two stages: offline data collection and online segmentation. During the offline stage, we collect the static background for training and pre-storage. Then, for each incoming frame, our online

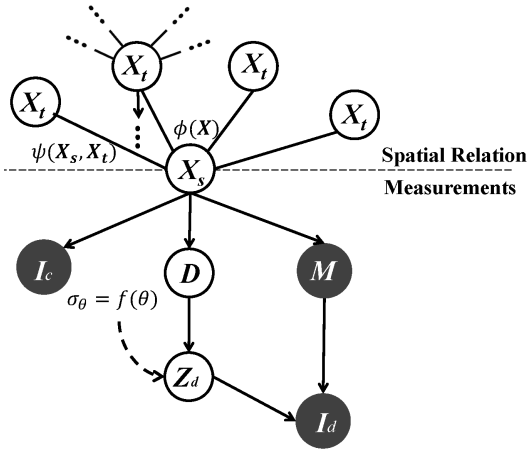


Fig. 3. Foreground-Background Depth Measurement Model

algorithm labels each pixel as either foreground or background via a probabilistic framework that incorporates background and measurement modeling as well as available observations in its neighborhood. In the following subsections, we describe the structure of the probabilistic frame and the specification of the model parameters.

A. Modeling of labels and RGB-D data

We describe the relationships among the foreground/background labels and the observations as a graphical model shown in Figure 3. Let G be the support of the 2D color and depth images. At each pixel location $s \in G$, X_s denotes the latent binary random variable indicating whether that pixel belongs to the background layer (-1) or the foreground layer (1). Spatially, it is connected to the four closest neighbors, generically referred as X_t. All the foreground/background labels over the entire image thus form a Markov Random Field (MRF) and the spatial relationship between adjacent labels is governed by an edge potential $\psi(X_s, X_t)$. Each foreground/background label X also has its evidence potential function $\phi(X_s)$ based on the measurement Bayesian Network (BN) shown in the lower half of Figure 3. As all the measurements are made at the same pixel, the subscript s is omitted and $\phi(X)$ is defined as follows:

$$\begin{aligned}\phi(X) &\triangleq \int_z \int_e P(X, D, Z_d, I_c, M, I_d)\, de\, dz \\ &= P(X) \cdot P(I_c|X) \cdot P(M|X) \cdot P(I_d|M, X) \quad (1)\end{aligned}$$

where

$$P(I_d|M, X) = \int_e P(D|X) \int_z P(Z_d|D) P(I_d|Z_d, M)\, dz\, de. \quad (2)$$

e and z represent the realizations of variables D and Z_d. Note that Equation (1) is defined as a marginal rather than a conditional as the normalization factor does not affect our MAP estimation described below. I_c represents the observed color values. D represents the true but unobserved depth values. D is corrupted by an additive Gaussian noise which produces a noisy measurement Z_d. Due to the missing depth problem, Z_d may not be directly observable. We thus introduce an observable indicator random variable M which is 1 if the depth value is observed and 0 otherwise. Combining these two random variables results in the observable depth value $I_d = MZ_d$. Using this probabilistic model, foreground-background labeling can be formulated as a MAP problem:

$$X_G^{map} \triangleq \arg\max_{x_G} \left(\sum_{s,\ t} \log \psi(x_s, x_t) + \sum_s \log \phi(x_s) \right) \quad (3)$$

where, s and t are the neighbor nodes in G. Our choice of parametrization allows $\psi(x_s, x_t)$ and $\phi(x_s)$ to be computed exactly. While the complexity of the exact solution to the MAP problem is exponential in the image size, approximate solution using loopy belief propagation can be easily obtained [17].

After assigning each pixel with either a foreground or background label, we need to identify erroneous depth measurements and complete missing depth values. The erroneous depth values are essentially outliers that are significantly different from other depth values in the neighborhood. However, as most measurement errors occur around object boundaries, it is imperative not to mistake true depth discontinuities as wrong depth values. The layer labels allow us to separate pixels that are likely to have come from objects at completely different depths. To determine if a depth pixel is an outlier, we robustly estimate the depth distribution in the neighborhood around the pixel via a RANSAC-like procedure. First, we only consider depth values in the neighborhood that share the same label as the target pixel. Then, multiple small sets of random sample pixels are drawn and a Gaussian distribution is estimated for each set. If only a small fraction of the neighborhood can be fit within two standard deviations from the mean of a sample distribution, this distribution is likely to contain outlier samples and is thus discarded. Among those that survive the robustness test, the one with the smallest variance is used and the target depth pixel is declared an outlier if it is beyond two standard deviations from the mean. The outlier depth pixel will join the rest of the missing depth pixels and will be completing using a joint color-depth bilateral filtering scheme similar to that in [23]. The only difference is that we only consider the contributions from neighboring depth pixels that have the same layer label as the center pixel.

B. Model Parametrization

We now describe the parametrization of the probabilistic model. For the spatial MRF, the edge potential $\psi(X_s, X_t)$ is defined based on the similarity in color and depth between the neighboring pixels:

$$\begin{aligned}\psi(X_s, X_t) &\triangleq \frac{1}{2} - X_s X_t \Big[\frac{1}{2} - \alpha E_c\left(I_c(s), I_c(t)\right) \\ &\quad -(1-\alpha) E_d\left(I_d(s), I_d(t)\right)\Big] \quad (4)\end{aligned}$$

The color similarity function $E_c\left(I_c(s), I_c(t)\right)$ is based on the cosine angle between the color channels:

$$E_c\left(I_c(s), I_c(t)\right) = \max\left\{\frac{I_c(s)^T I_c(t)}{\parallel I_c(s) \parallel \cdot \parallel I_c(t) \parallel},\ n_f\right\} \quad (5)$$

where n_f is the noise floor used to prevent zero potential. The depth similarity function is just the absolute difference:

$$E_d\left(I_d(s), I_d(t)\right) = \max\left\{1 - \frac{\mid I_d(s) - I_d(t) \mid}{D_{max}},\ n_f\right\} \quad (6)$$

where D_{max} is the dynamic range of the depth measurement. $\alpha \in [0, 1]$ is an important parameter controlling the relative weights assigned to color and depth: if the depth measurements are reliable, most of the weight should be assigned to depth values as they are more reliable for foreground/background labeling; if the depth measurements are unreliable, they should not be used at all in computing the edge potential. As argued in Section 1, the uncertainty in the depth measurement depends on many factors. As erroneous depth measurements occur predominantly near object boundaries, we apply an edge detector on the depth map and use the spatial distance θ to the closest depth edge as a reliability measure. α is defined as the logistic function of θ:

$$\alpha \triangleq f\left(\tfrac{\theta_s + \theta_t}{2}\right) \text{ with } f(\theta) \triangleq \tfrac{1}{1+e^{-(\tau-\theta)}} \quad (7)$$

τ is a distance threshold beyond which the depth value is relatively noise free. This simple model is easy to compute, though a more sophisticated one incorporating surface normal, texture, and color can be used in a similar fashion.

For the measurement BN, we assume $P(X)$ is uniform and model the background color distribution $P(I_c|X=-1)$ as a MOG distribution trained by a video sequence of the static background using the Codebook technique [18]. The foreground color distribution $P(I_c|X=1)$ is assumed to be uniform over the entire range. The background depth distribution $P(D|X=-1)$ is modeled as a single Gaussian distribution $\mathcal{N}(\mu_{d,-1}, \sigma^2_{d,-1})$ estimated from the static background sequence. The foreground depth distribution $P(D|X=1)$ is also modeled as a single Gaussian $\mathcal{N}(\mu_{d,1}, \sigma^2_{d,1})$. As the foreground is closer to the camera (smaller depth values) than the background, $\mu_{d,1}$ is set to be the midpoint between 0 and the background mean $\mu_{d,-1}$ and $\sigma_{d,1}$ is half of $\mu_{d,1}$. The noisy depth measurement Z_d is modeled based on an additive Gaussian model:

$$Z_d \triangleq D + N, \quad \text{with } N \sim \mathcal{N}(0, \sigma^2_\theta) \text{ and } N \perp D. \quad (8)$$

The noise standard deviation σ_θ reflects the uncertainty in the depth measurement. Similar to our earlier approach in determining the depth weight for the edge potential, we use $\sigma_\theta \triangleq \mathcal{C} f(\theta)$ where $f(\theta)$ is the logistic function defined in (7) and $\mathcal{C}$ is a constant scaling parameter.

As described in Section 1, missing depth values occur mostly in the background region due to the disparity between the IR projector and IR camera. Our empirical experiments show that there are about three times as many background pixels as foreground pixels for those with missing depth measurements, but stay roughly the same for those with valid depth measurements. As such, we set $P(M=0|X=-1) = 0.15$ and $P(M=0|X=1) = 0.05$. The actual depth measurement $I_d = MZ_d$ implies that $P(I_d = i_d|Z_d = z, M = m) = \delta_{mi_d}(z)$, the dirac delta function with the only non-zero value at $z = mi_d$. Substituting this into (2) results in the following simplification:

$$\begin{aligned} & P(I_d = i_d | M = m, X = x) \\ &= \int_e P(D = e|X = x) P(Z_d = mi_d | D = e)\, de \\ &= P(Z_d = mi_d | X = x) \end{aligned} \quad (9)$$

Given $X = x$, Z_d and D are multivariate with the following distribution:

$$\begin{bmatrix} Z_d \\ D \end{bmatrix} \Big| X = x \ \sim\ \mathcal{N}\left(\begin{bmatrix} \mu_{d,x} \\ \mu_{d,x} \end{bmatrix}, \begin{bmatrix} \sigma^2_{d,x} + \sigma^2_\theta & \sigma^2_{d,x} \\ \sigma^2_{d,x} & \sigma^2_{d,x} \end{bmatrix}\right) \quad (10)$$

Thus, $Z_d|X = x \ \sim\ \mathcal{N}(\mu_{d,x}, \sigma^2_{d,x} + \sigma^2_\theta)$ and (9) can be numerically evaluated.

V. Experimental Results

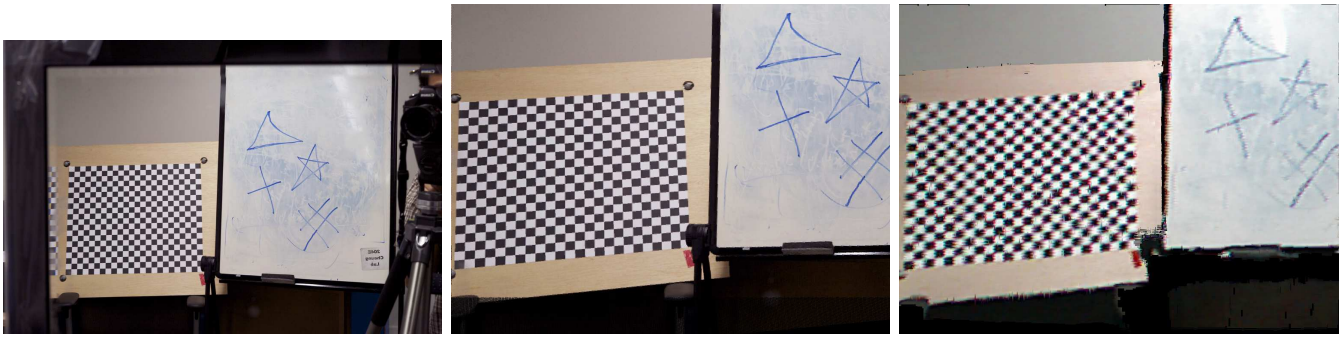
In this section, we first show the simulation results to evaluate our virtual mirror model. Then, we compare the quality of rendered images generated by our proposed depth denoising scheme and other denoising schemes in the literature. Finally, we present some performance measurement of a preliminary implementation of a complete system.

A. Rendering accuracy

In order to validate the accuracy of our mirror model, we compare the rendered 1078×786 mirror image I_1 on the display and a real mirror image I_2 taken by a digital camera looking at a real mirror aligned with the display. The camera is in the same position as the asserted viewpoint that the mirror system uses for rendering.

To compare the virtual mirror image with the real mirror image, we align the two images by estimating the homography between them. Here we use the four corners of the real mirror from the picture to match the four corners of virtual mirror image represented as: $p_1 = (0\ 0)^T$, $p_2 = (0\ 1024 \times \frac{w_2}{w_1})^T$, $p_3 = (0\ 768 \times \frac{h_2}{h_1})^T$ and $p_4 = (1024 \times \frac{w_2}{w_1}\ 768 \times \frac{h_2}{h_1})^T$. w_1, h_1 and w_2, h_2 denote the size of the display and real mirror respectively. With these four pairs of corresponding points, it is sufficient to compute homography matrix H and the aligned images are shown in Figure 4.

To measure the rendering accuracy, 100 corner points are manually selected on I_1 and I_2 for analysis. To ensure that the exact pixel locations of the corners are used, we compute the *Normalized Cross Correlation*(NCC) over a 3×3 neighborhood around each corner point and refine the corner position to that with the maximum local NCC value. All the matching pairs of corner points are shown in Figure 5. The position differences between the matching pair are indeed quite small – the mean is 1.4865 pixel with standard deviation 0.8156.



(a) Real Mirror Image (b) Homography Transformation on the 4 corners of the mirror (c) Generated Virtual Image

Fig. 4. Compare our virtual mirror with a real mirror

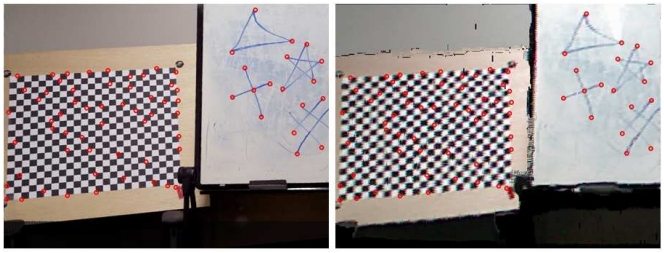


Fig. 5. *Matching Points on Both Images*

B. Depth Improvement Evaluation

Figures 6(a) and 6(b) show a sample of RGB and depth images captured by a Kinect. A significant portion of the depth measurements around the foreground person is missing. The missing region straddles both foreground and background, though the majority of the missing pixels are within the background. There are also erroneous depth values, most notably around the fingers and the hair. Figure 6(c) shows the layer mask obtained by our algorithm. The contour of the foreground object is perfectly recovered. This mask is better than those obtained through background subtraction on either the depth image where the contour is too noisy, or the color image where the blue shirt on the person is mistaken for the blue box background. Guided by this segmentation mask, the corrected and completed depth image by our method is shown in Figure 6(d).

For comparison, we first investigate if background modeling alone is sufficient to correct the depth values. Figure 6(e) shows the result of filling in any missing depth value by the mean of the trained background distribution at that pixel. The assumption that all missing depth values come from the background results in wrong depth values in part of the hair, the fingers and the body. The second scheme that we compare is a simplified version of [19] in which missing depth values are interpolated by surrounding depth pixels with similar color via a bilateral filter. The result is shown in Figure 6(f). Despite a smoother and better defined foreground contour, this approach enlarges the foreground shape especially around the right arm due to the similarity in color between the foreground shirt and the background box. The contour of the fingers is also poorly rendered because of the erroneous depth values in that region. The differences among these two schemes and ours are even more dramatic when we render mirror-like virtual image. Figures 6(g), 6(h), and 6(i) depict the rendered views of our scheme, the background replacement and the color-based depth in-painting respectively. Our scheme clearly produces the best rendering with all the newly revealed background behind the person filled in by the background model. While the background replacement scheme can also do that, the erroneous depth values leave residual details around the fingers and the hair. The color-based depth in-painting is unable to fill in any revealed background. Also, the wrongly assigned depth values move some of the background pixels with the foreground along the right side of the head, while some pixels of the fingers are stuck at the background.

C. Virtual Mirror System Experiment

We have a preliminary prototype mirror system with two Kinect implemented using C++ and OpenCV library. Each Kinect captures 640×480 resolution video for scene points generation and the local client renders the virtual image with resolution 1024×768, which is the same size as the final image on the server. The server and client machines are as follows:

- **Server** : Intel Xeon E5335 processor with 4-core CPUs at 2.0 GHz and 4.0Gb of RAM.
- **Client** : Intel Core(TM) E8400 Duo CPU at 3.00 GHz and 8.0Gb of RAM

No special software optimization or hardware acceleration is currently employed. Without the depth denoising algorithm, our system can render roughly three frames per second. Due to the significant complexity of the iterative loopy belief propagation, the system needs roughly 4 seconds to render one frame when the depth denoising algorithm is applied.

VI. Conclusions

In this paper, we have presented a framework for rendering virtual mirror images by fusing multiple RGB-D cameras. The

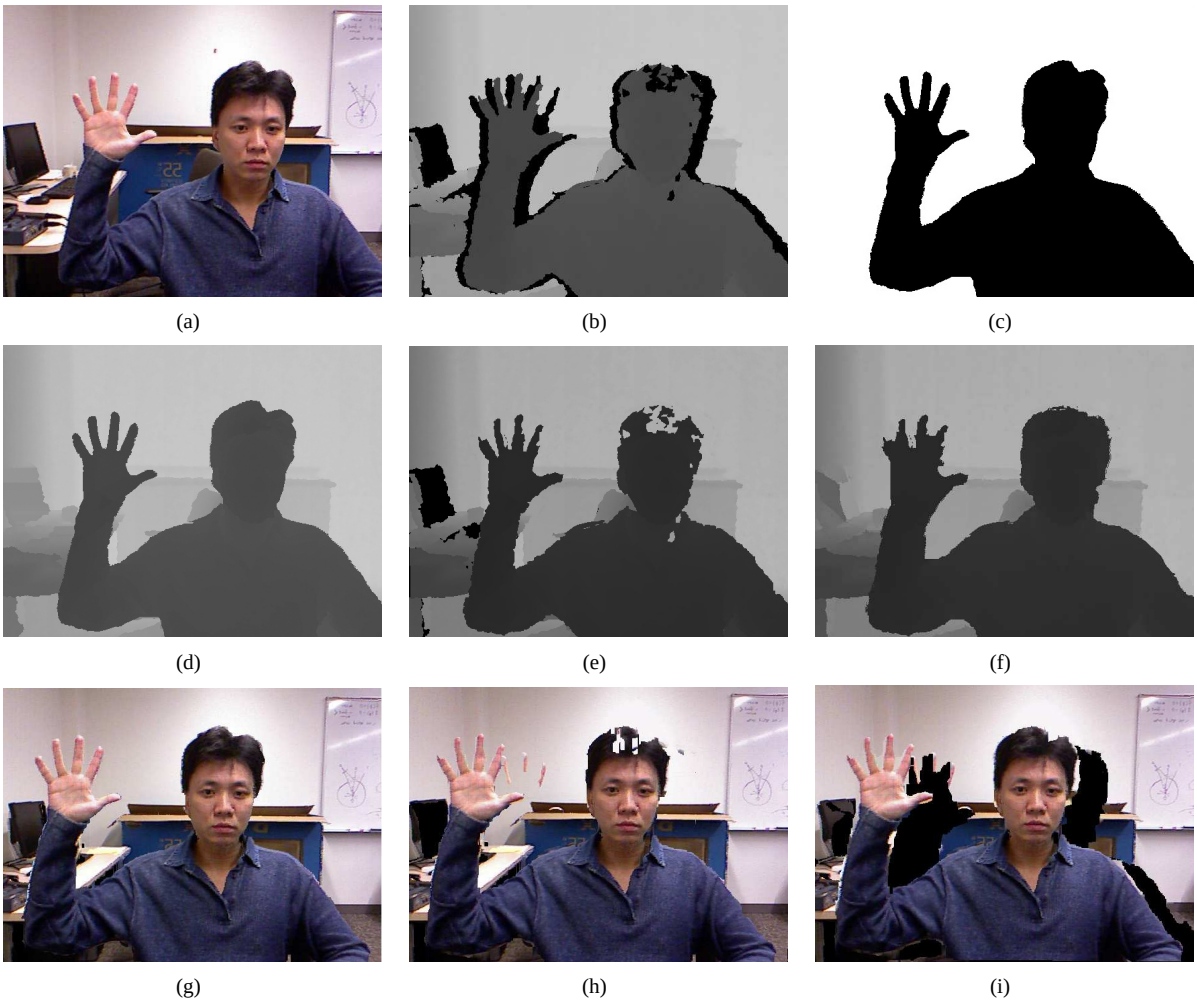


Fig. 6. (a) raw RGB frame; (b) raw depth image; (c) layered mask; (d), (e) and (g): completed background by our proposed scheme, background replacement and color-based depth in-painting respectively; (g), (h) and (i): virtual view by the same three schemes.

initial depth data has been thoroughly enhanced using a depth denoising algorithm. Our depth denoising algorithm takes advantage of a novel probabilistic background/foreground separation to eliminate outliers and complete missing values. Once the depth data are cleansed, we have shown that they can be used to estimate the viewpoint and can be locally aligned to create a 3D point cloud to render a viewer-dependent mirror image. The server then aggregates all the partially rendered mirror images to compute the final result. Our current implementation does not meet the real-time requirement of a virtual mirror system. However, it is anticipated that much of the rendering pipeline can be significantly accelerated with the use of GPUs. The ultimate bottleneck will likely reside in the iterative belief propagation step. We observe that the MRF results do not change significantly from frame to frame. This suggests that significant speedup could be achieved by replacing the multiple-round belief propagation algorithm with a local update procedure.

REFERENCES

[1] J. Ehara and H. Saito. Texture overlay for virtual clothing based on pca of silhouettes. In Proceedings of the 5th IEEE and ACM International Symposium on Mixed and Augmented Reality, ISMAR 06, pages 139142, Washington, DC, USA, 2006. IEEE Computer Society. 1, 2.

[2] P. Eisert, P. Fechteler, and J. Rurainsky. 3-D Tracking of Shoes for Virtual Mirror Applications. In Proceedings of the International Conference on Computer Vision and Pattern Recognition (CVPR2008), Anchorage, Alaska, USA, 24-26th June 2008. CVPR 2008. 1, 2.

[3] A. Taguchi, T. Aoki, and H. Yasuda. A study on real-time virtual clothing system based on two-dimensional plane model. In Information and Telecommunication Technologies, 2005. APSITT 2005 Proceedings. 6th Asia-Pacific Symposium on, pages 126 130, nov. 2005. 1, 2.

[4] V. Ramachandran, D. C. Rogers-Ramachandran, and S. Cobb. Touching the phantom. Nature, 377:489490, 1995. 1.

[5] V. S. Ramachandran and E. L. Altschuler. The use of visual feedback, in particular mirror visual feedback, in restoring brain functions. Brain, 132:16931710, 2009. 1.

[6] C. for Disease Control and Prevention. Autism spectrum disorders - data and statistics, May 2010. 1.

[7] L. Uddin, M. S. Davies, A. A. Scott, E. Zaidel, S. Y. Bookheimer, M. Lacoboni, and M. Dapretto. Neural basis of self and other representation in autism: An fmri study of self-face recognition. PLoS ONE, 3(10), 2008. 1.

[8] T. Buggey. Seeing Is Believing: Video Self Modeling for people with Autism and other developmental disabilities. Wodbine House, 2009. 1.

[9] M. Biehl, A. Ghosh, and B. Hammer. Learning vector quantization: The dynamics of winner-takes-all algorithms. Neurocomput., 69:660670, March 2006. 2, 5.
[10] V. Kitanovski and E. Izquierdo. 3d tracking of facial features for augmented reality applications. In International Workshop on Image Analysis forMultimedia Interactive Services, Delft, The Netherlands, 2011. 2.
[11] C. Cullinan and S. Agamanolis. Reflexion: a responsive virtual mirror for interpersonal communication. In European Conference on Computer Supported Cooperative Work, 2003. 2.
[12] A. R. J. Francois and E.-Y. E. Kang. A handheld mirror simulation. In Proceedings of the 2003 International Conference on Multimedia and Expo - Volume 1, ICME 03, pages 745748, Washington, DC, USA, 2003. IEEE Computer Society. 2
[13] D. Porquet, J.-M. Dischler, and D. Ghazanfarpour. Real-time highquality view-dependent texture mapping using per-pixel visibility. In Proceedings of the 3rd international conference on Computer graphics and interactive techniques in Australasia and South East Asia, GRAPHITE 05, pages 213220, New York, NY, USA, 2005. ACM. 2.
[14] C. Eisenacher, Q. Meyer, and C. Loop. Real-time view-dependent rendering of parametric surfaces. In Proceedings of the 2009 symposium on Interactive 3D graphics and games, I3D 09, pages 137143, New York, NY, USA, 2009. ACM. 2.
[15] P. Debevec, Y. Yu, and G. Boshokov. Efficient view-dependent image-based rendering with projective texture-mapping. Technical report, Berkeley, CA, USA, 1998. 2.
[16] A. Z. Richard Hartley. Multiple view geometry in coputer vision, 1984. Supplied as additional material tr.pdf. 6.
[17] Freeman, W., Pasztor, E., Carmichael, O.: Learning low-level vision. International journal of computer vision 40 (2000) 2547
[18] Harville, M.: A framework for high-level feedback to adaptive, per-pixel, mixture-ofgaussian background models. Computer VisionECCV 2002 (2002) 3749
[19] Garro, V., dal Mutto, C., Zanuttigh, P., Cortelazzo, G.: A novel interpolation scheme for range data with side information. In: VisualMedia Production, 2009. CVMP09. Conference for, IEEE (2009) 5260
[20] Khoshelham, K.: Accuracy analysis of kinect depth data. In: ISPRS Workshop Laser Scanning. Volume 38. (2011).
[21] Yang, Q., Yang, R., Davis, J., Nister, D.: Spatial-depth super resolution for range images. In: Computer Vision and Pattern Recognition, 2007. CVPR07. IEEE Conference on, IEEE (2007) 18
[22] Wang, L., Jin, H., Yang, R., Gong, M.: Stereoscopic inpainting: Joint color and depth completion from stereo images. In: Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on, IEEE (2008) 18
[23] Kopf, J., Cohen, M.F., Lischinski, D., Uyttendaele, M.: Joint bilateral upsampling. ACM Transactions on Graphics (Proceedings of SIGGRAPH 2007) 26 (2007)