Shadow Removal in Front Projection Environments Using Object Tracking

Samuel Audet and Jeremy R. Cooperstock
Centre for Intelligent Machines, McGill University
3480 University Street, Montréal, Québec, Canada
{saudet,jer}@cim.mcgill.ca

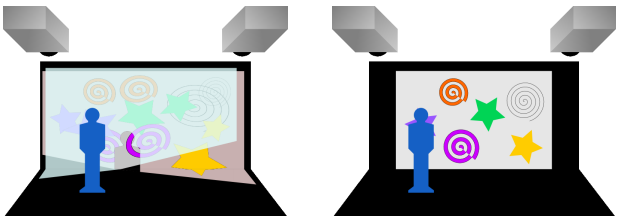
Abstract

When an occluding object, such as a person, stands between a projector and a display surface, a shadow results. We can compensate by positioning multiple projectors so they produce identical and overlapping images and by using a system to locate shadows. Existing systems work by detecting either the shadows or the occluders. Shadow detection methods cannot remove shadows before they appear and are sensitive to video projection, while current occluder detection methods require near infrared cameras and illumination. Instead, we propose using a camera-based object tracker to locate the occluder and an algorithm to model the shadows. The algorithm can adapt to other tracking technologies as well. Despite imprecision in the calibration and tracking process, we found that our system performs effective shadow removal with sufficiently low processing delay for interactive applications with video projection.

1. Introduction

Front projection installations use less space than equivalent rear projection configurations and cost less than monitors to cover large display surfaces. However, when a person moves in between a projector and a display surface, a shadow occurs, negatively affecting the experience in terms of human-computer interaction [14].

In many situations, we can manage shadows by appropriately positioning projectors and constraining the allowed locations at which people can stand. When this is not possible, such as in small rooms, we can use techniques where multiple projectors are positioned far apart and configured so they produce the same image at the same place on the display surface [13]. This is also known as *Passive Virtual Rear Projection*. The image thus remains visible even if a person occludes one projector. This configuration can be achieved by transforming the images before their projection so they match with the reference frame [11, 18]. For a flat display surface, the transformation may be a simple homography. Color correction and multi-focal projection are also desirable, although outside the scope of our work.



(a) Before corrections. (b) After corrections.

Figure 1. Ideal front projection where distortion, colors and shadows are all corrected.

With the previous approach, although the image remains visible, the resulting differences in display intensity remain perceptible. To compensate, more advanced methods compute and determine the display region in which a shadow occurs. Another projector can then fill the region with equal intensity and identical color to that of the occluded projector, as in the ideal case depicted in Figure 1. This is also known as *Active Virtual Rear Projection*. However, as described below, current methods suffer from a number of limitations.

1.1. Related Work

The first methods that were developed directly detect shadows on the display surface [3, 6, 8, 13] and thus can only remove shadows after they appear. Visual echo may also result when the algorithm no longer observes a shadow and incorrectly assumes the region to be unoccluded. This method works by comparing the image displayed by a projector to the image as seen by a camera, after compensating for geometric and radiometric color differences between the projector and the camera. Therefore, this operation is subject to numerous errors in the case of video projection, since to compare correctly we need to know the precise delay incurred in the projectors, in the cameras, and in processing. An additional constraint is that the cameras need an unoccluded view of the entire display surface, or the system cannot function properly.

Other methods use a near infrared camera mounted

alongside each projector [4, 17] to detect the occlusion rather than the shadow. Because these cameras do not see projector light, a simple background subtraction technique can be used to generate a pixel-mask of the occlusions in front of each camera-projector pair. Since background subtraction requires good contrast between the background (display surface) and the foreground (people), infrared floodlights are used to illuminate the display surface, while not illuminating the people. However, this method, on its own, cannot predict where a shadow will appear next as people move within the environment. To deal with this limitation, Flagg *et al*. [4] perform much of their computation on Graphics Processing Units (GPUs), which greatly improves processing speed and the results. The main drawback of this approach is the requirement of hardware specifically for shadow removal purposes, which may include infrared floodlights, and for each projector, a near infrared camera and additional processing power (GPUs).

1.2. Object Tracking for Shadow Removal

Instead, we propose object tracking as the basis of our approach to shadow removal. Many interactive applications already require such tracking, for example, to ensure that images are rendered from the proper perspective [12] or to adjust 3D audio according to user movement [20]. This leverages technology that may already be in place, since our method can use data from any object tracker. Also, tracking can take advantage of temporal information to predict the motion of people as occluders.

Our approach requires a calibration method for both the cameras and projectors, an adequately reliable object tracker, and an algorithm that combines this information to perform shadow removal. While calibration and tracking are often imprecise, we show that good results can be obtained from such a system.

A high-level diagram of our system architecture appears in Figure 2. Our work focuses on the Calibration, Object Tracking, Shadow Removal, and Distortion and Perspective Correction modules. We did not address issues of color correction and multi-focal projection. In the following sections, we elaborate on our approach. Section 2 first discusses the camera calibration procedure we employed, then describes how we adapted it to projector calibration. Next, Section 3 describes how we adapted a camera-based object tracker for our purposes. The main challenge we faced was to achieve reasonably reliable tracking, despite the projection of dynamic video in the background, which can easily be misinterpreted as moving people. Section 4 explains how we integrated these components to model shadows and achieve shadow removal. Section 5 summarizes our experimental results and Section 6 concludes with a discussion of limitations and future work.

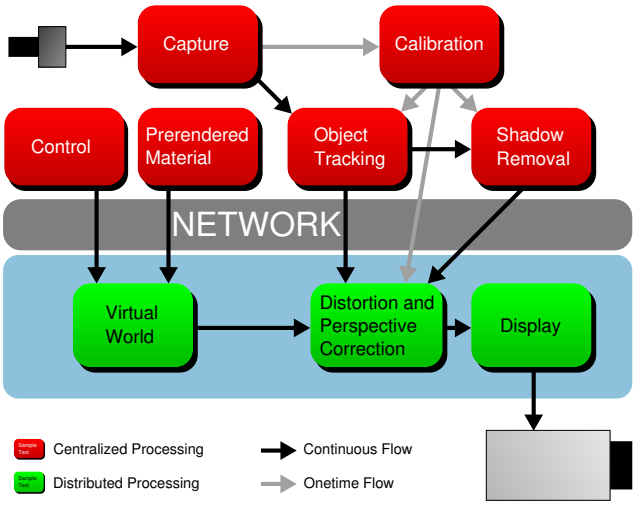


Figure 2. Illustration of the flow of data in our system.

2. Calibration

We first needed to find the intrinsic and extrinsic parameters of both the cameras and the projectors. These parameters are required to obtain the location and size of people in 3D and to model their shadows occurring on the display surface.

2.1. Intrinsic Parameters

To find the intrinsic parameters of our *perspective devices* (cameras and projectors), we used Zhang's calibration method [21] as implemented by Bouguet [2]. This method was designed for camera calibration, so we used it directly to calibrate our cameras. For the projectors, we developed a method based on work done by Raskar and Beardsley [10], and by Ashdown *et al*. [1]. Their methods involves locating projected points on a physical board using the homography between the projector and a camera viewing the board. In the second paper, the red and blue channels of a color camera were used respectively as filters for a calibration board composed of white and cyan squares, and for a projected pattern with blue and black squares. This way, one can use the same surface for both a projected pattern and a physical pattern, maximizing the use of the sensor of a color camera, so we chose this method. We tested cyan, magenta, yellow, red, green and blue colors under complementary projector light, and we also came to the conclusion that cyan was the best color to use. During calibration of a projector, since we did not have mobile projectors as Ashdown *et al*. [1], we held our cyan calibration board, a printed sheet attached to a foam board, at different angles in front of the projector as depicted in Figure 3. We found that our projectors were too powerful and they emitted too much blue light in comparison to the ambient red light. Rather than simply reducing the intensity of the projector, we took advantage of

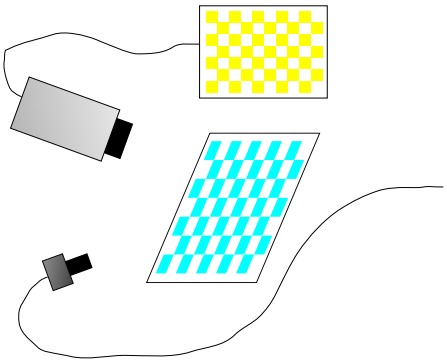


Figure 3. Holding the cyan calibration board in front of a camera and a projector displaying the yellow pattern.

the opportunity. We used the projector to light up the calibration board with red and green light as well, resulting in a projected pattern with yellow and white squares. This way, we were able to guarantee the ratio of blue pixel intensity to red pixel intensity without relying on the environment or fine-tuning camera parameters. Typical images are shown in Figure 4.

We also took care of other important factors that can influence the precision of the resulting calibration. Previous research [16] indicates that to obtain near optimal results with the method of Zhang [21], the calibration board should have more than 150 corners and the set of images should contain more than 1500 corners. For these reasons we used at least 15 images for the calibration of each device, and the calibration board used for the cameras had 18×14 squares of 15 mm (221 corners), the cyan calibration board had 19×14 squares of 50 mm (234 corners), and the projected pattern had 16×12 squares (165 corners). Also, we held the calibration boards at angles near 45°as recommended by Zhang [21]. Last, when calibrating a projector, we installed the camera on top of the projector so that the projected pattern image would use the largest possible area of the camera sensor even when holding the board at different angles.

2.2. Extrinsic Parameters

To obtain the extrinsic parameters of the cameras and projectors, we first found the homographies induced by the flat display surface for each camera-projector pair by projecting a known calibration pattern onto the surface. For two calibrated devices, an algorithm developed by Triggs [19] can then decompose the homography into the rotation and translation separating the two devices. It also provides the equation of the inducing plane. The algorithm actually returns two sets of values, only one of which corresponds to the physical reality. In our case, the display surface remained the same for at least two projectors, so we could easily resolve the ambiguity and retain the set of values for which the plane equations were closest to one another.

Because of unavoidable calibration errors in the intrinsic parameters, the plane equations were not actually equal. To model the shadows on the display surface, a unique plane equation is required to properly backproject from two or more devices. Also the homographies between projectors were very precise (less than one camera pixel of error), and we did not want to lose this precision. To satisfy both goals, we devised the following procedure. First, we decided to use a plane equation $\mathbf{p}_\mathrm{c}$ found with the homography between two cameras, since it was probably more accurate given that the calibration errors for the projectors were larger. Then, the idea is to find a corrective homography H to remove the discrepancies between the projection of 3D points $\mathbf{X}$ and known corner locations $\mathbf{x}_\mathrm{p}$, which should be equal on the image plane of the projector:

$$\mathbf{x}_\mathrm{p} = \mathrm{H}\mathrm{P}_\mathrm{p}\mathbf{X} \tag{1}$$

where

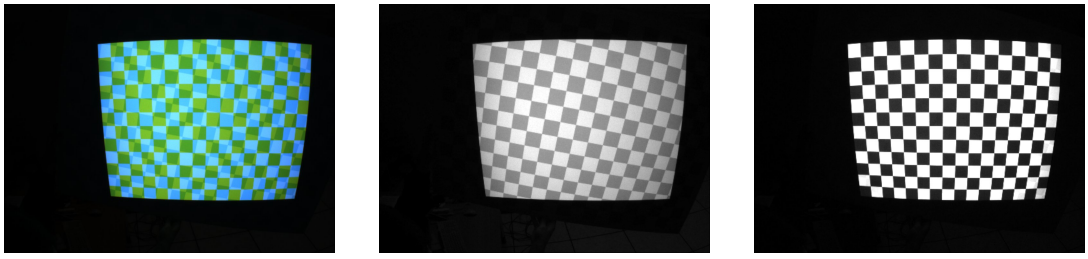
$$\mathbf{X} = \mathrm{B}_\mathrm{c}\mathbf{x}_\mathrm{c}\,, \tag{2}$$

$$\mathrm{B}_\mathrm{c} = [\mathbf{p}_\mathrm{c}\mathbf{t}_\mathrm{c}\mathrm{I}_{4\mathrm{x}4} - \mathbf{t}_\mathrm{c}\mathbf{p}_\mathrm{c}]\mathrm{P}_\mathrm{c}^{+}\,, \tag{3}$$

$$\mathrm{P}_\mathrm{c}^{+} = \mathrm{P}_\mathrm{c}^{\mathrm{T}}(\mathrm{P}_\mathrm{c}\mathrm{P}_\mathrm{c}^{\mathrm{T}})^{-1} \tag{4}$$

where $\mathbf{X}$ are the 3D location of the points on the display surface $\mathbf{p}_\mathrm{c}$ according to their projection $\mathbf{x}_\mathrm{c}$ in the camera and its backprojection matrix B_c where $\mathbf{t}_\mathrm{c}$ is the camera center and $\mathrm{P}_\mathrm{c}^{+}$ is the *pseudoinverse* of the projection matrix P_c of the camera. Hartley and Zisserman [5] provide more detailed explanations on this subject. H is the corrective homography we are looking for so that once added to the projection matrix P_p of the projector, $\mathbf{X}$ project to $\mathbf{x}_\mathrm{p}$ as closely as possible with least squares. We ignored the radial and tangential distortion coefficients of the projector as found during calibration. This approximation should be valid as the projectors did not exhibit noticeable distortion. Also, this simple procedure as a whole does not attempt to minimize geometric errors induced inside the projection matrix, but assuming relatively precise calibration results, the correction, and consequently the errors, should be minimal.

Finally, to obtain equation of the floor plane required by the tracking module as detailes below, we manually located the edge vector in the camera image where the floor and the display surface met. We also estimated the length of the vector in a physical unit. Assuming the floor and the surface were at right angles, we found the normal of the floor by computing the cross-product of the normal of the surface and of the edge vector. We also rescaled all calibration parameters according to the physical length of the vector.



(a) RGB image. (b) Red channel. (c) Blue channel.

Figure 4. Sample images we used for projector calibration. (We adjusted the level of the red channel for better contrast.)

3. Tracking

In theory, we could use any tracking algorithm that provides 3D information. However, we wanted to test our system with an algorithm that did not require the user to wear sensors, tags or markers, and that could also work in an environment filled with video projections. We devised a simple method based on previous research using computer vision [7, 15]. We did not focus our efforts into developping a new tracking method, and although we could have used more complex methods for better 3D tracking, we were motivated in using an approach we could easily implement and that would be fast enough for interactive purposes.

3.1. Disparity Contours

First, we used the calibration data of the room (floor and display surface) to build a background disparity map. This map can be seen as a function $\mathbf{x}_2 = M(\mathbf{x}_1)$ that maps points $\mathbf{x}_1$ from the image plane of one camera into points $\mathbf{x}_2$ on the image plane of the other camera. As detailed by Ivanov *et al*. [7], with a pair of frames I_1 and I_2 and their disparity map, one can compute a difference image

$$\mathrm{D}(\mathbf{x}_1) = |\mathrm{I}_1(\mathbf{x}_1) - \mathrm{I}_2(M(\mathbf{x}_1))| \tag{5}$$

for all points $\mathbf{x}_1$. In theory, $\mathbf{x}_1$ and $M(\mathbf{x}_1)$ should backproject to the same physical point when no object is present in front of the cameras, thus producing a completely black difference image D regardless of changes in lighting conditions. However, when the disparity map does not correspond to the geometry of what is actually present in front of closely positioned cameras, this procedure produces contour images, referred to as disparity contours after Sun [15], similar to the results of an edge detector. Although in theory it is possible to extract depth information from those contours, the task has proven challenging [15], and we instead used the information for 2D tracking only.

We decided to apply to the resulting difference image a 3×3 square mask erosion, to deal with ± 0.5 pixel registration errors, followed by a background subtraction algorithm using a Mixture of Gaussian [9], which worked best at minimizing the importance of the various causes of errors. We then routed the processed images to the blob tracker. On the output of the tracker, we used a Kalman filter to predict movement. The filter runs two prediction steps, accounting for the delays in the camera/tracker module and in the post-processing/projector module, assuming they are close. In this manner, we obtained an object tracker insensitive to dynamic video projection on the display surface as showed in Figure 5.

3.2. 3D Tracking

The tracker only provides 2D information, but we used the following method to recover 3D information by assuming that the objects of interest would stand vertically on the floor, which is usually the case with people. With equation 2, we can backproject a point $\mathbf{x}_\mathrm{c}$ corresponding to the feet of a person in the image plane of the camera into a point $\mathbf{X}$ on the floor whose normal is known. We can then approximate the height of the person by considering a plane parallel to the floor on top of the person $\mathbf{p}_\mathrm{top} = [\ a \quad b \quad c \quad e\ ]$, which consequently has the same normal as the plane of the floor $\mathbf{p}_\mathrm{floor} = [\ a \quad b \quad c \quad d\ ]$, where the normal is $\mathbf{n} = [\ a \quad b \quad c\ ]^T$. The distance we are interested in is thus the height $h = d - e$. Looking back at equation 2, the backprojection matrix B_c has rank 3, so it can solve for 3 unknowns. Our unknowns are the height h and the scale of the resulting $\mathbf{X}$. The system is thus overdetermined. We could have used least squares to minimize errors with $\mathbf{x}_\mathrm{c} = [\ u \quad v \quad 1\ ]$, but under the assumption that the up vector of the camera is in the general direction of the nor-

(a) Camera image. (b) Disparity contours and tracking.

Figure 5. Sample camera image and the corresponding disparity contour image with tracking information.

mal, simply dropping the u component and using only the v component should produce acceptable results, so this is what we did. In this manner, we decided to model people as flat rectangles parallel to the display surface. For the current implementation of the algorithm, this simplifying assumption appeared sufficient.

4. Shadow Removal

Our system runs the tracker and generates masks for all projectors as a centralized process. On the other hand, the post-processing of the masks just before display can be done in parallel and be distributed.

4.1. Centralized Processing

Shadow removal works by creating a rectangular mask image for each of the n projectors in a centralized manner. The area of the mask image represents the rectangular portion of the display surface where we desire images to be displayed. For the sake of clarity, we will use $[0,\ 1]$ as the range of pixel values for discussion in this section, but our implementation appropriately scales the results for use with 8-bit grayscale images. After processing, a pixel with value of 0 means that no light should be emitted for this pixel, and 1, the full brightness of the projector should be used for this pixel.

First, the algorithm sets all the pixels of the mask images to 0. Then using the calibration information it sets to 1 the pixels inside the quadrilateral region that is coverable by the projector. At this point, the images contain only information about which area of the display surface any particular projector can cover. Using tracking and calibration information, it then localizes shadows on the display surface. We can model shadows similarly to how the operation is done in computer graphics. For each projector, we can backproject the vertices of the rectangle onto the display surface, and draw with 0 valued pixels in mask images the resulting shadow. Finally it normalizes each pixel so that the sum of all pixels in the same position from all masks M does not exceed 1, as described by the following equation:

$$\hat{\mathrm{M}}_j(u,v) = \begin{cases} 0 & \text{if } \sum_{k=1}^{n} \mathrm{M}_k(u,v) = 0 \\ \frac{\mathrm{M}_j(u,v)}{\sum_{k=1}^{n} \mathrm{M}_k(u,v)} & \text{otherwise} \end{cases} \quad (6)$$

for $j = 1..n$ where M_j is the mask image before normalization, and $\hat{\mathrm{M}}_j$ after, for all pixels (u, v). However, when an occluder stands too close to the display surface, a region of the display surface will have all its pixels set to 0 for all projectors. To regain any small portion that is not actually occluded, the algorithm resets to 1 all pixels of the first projector in this area.

4.2. Distributed Processing

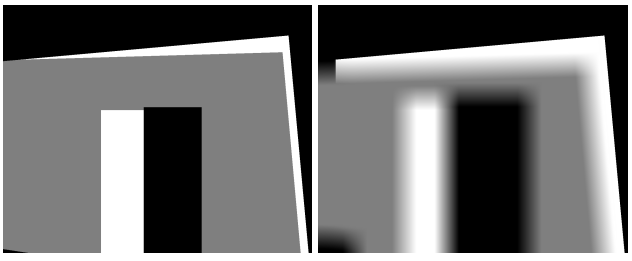
The masks are then dispatched over the network and used to modulate the brightness of the pixels of the projectors. First, to soften the transition between projectors with different color emission properties, we added a smoothing stage. We adapted the box blur algorithm to dynamically select the direction of the blurring and in this manner to try to clamp the pixel values in two kinds of regions of the masks: the shadow regions and regions near the edges of coverable areas. Figure 6 shows the difference before and after blurring. Next, the linear representation of intensity was not physically reproduced by our projectors. Since video systems usually follow a power law of 2.2, we performed gamma correction, *i.e.*: $\hat{\mathrm{M}}'(u,v) = \hat{\mathrm{M}}(u,v)^{\frac{1}{2.2}}$ for all pixels (u, v). Finally, the distributed program also corrected for linear (perspective) and non-linear (radial and tangential) distortion by approximating the desired correction function using OpenGL textures [3, 18]. Tardif *et al*. [18] used a grid of 12065 squares, so we divided our texture in a similar manner, more precisely into 101×101 vertices.

5. Results

We tested our system using front projection on an interior wall surface. We used two Sanyo PLC-EF30 projectors and two Point Grey Research Flea2 cameras equipped with 4.0-12.0 mm varifocal lenses adjusted at their widest angle. These cameras automatically synchronize with each other. The experiments involved three distinct steps; calibration, tracking, and shadow removal, each of which is described in detail below.

5.1. Calibration

Based on our modified version of Bouguet's calibration software [2], the reprojection errors (three times the standard deviation in pixels) achieved after calibration of the cameras and projectors are shown in Table 1. We performed the reprojection using the native resolution of the device as defined during calibration. The resolution of the cameras was 1024×768, and that of the projectors was 1280×1024.



(a) Before blur. (b) After blur.

Figure 6. Sample result of our modified box blur for smooth transition between two or more projectors.

During projector calibration, for consistency, we used images and calibration results from Camera 1 only.

	Reprojection error	
	u	v
Camera 1	0.274	0.250
Camera 2	0.251	0.212
Projector 1	0.724	0.777
Projector 2	0.499	0.543

Table 1. Reprojection error (three times the standard deviation in pixels) after calibration.

As for the extrinsic parameters, Table 2 details the position and orientation in space as found by our method. The normal of the floor was (0, 0, 1), while the normal of the wall was (0, -1, 0), and the up vector of the perspective device reference system was (0, 0, 1) while its viewing or projecting direction was (0, 1, 0), all axes following the right-hand rule convention. After applying corrective homographies, the errors between corresponding points from cameras or projectors all dropped to less than one pixel, as expected, according to the initial homographies. Interestingly enough, the corrective homographies did not significantly alter the rotations and translations of the projection matrices, preferring to skew the internal parameters instead. Although the homographies are close to the identity matrix, the operation minimizes the algebraic error, not the geometric error. Without proper normalization, it is not surprising to see such results. Nevertheless, our implementation performed well even with these errors.

	Position (cm)			Orientation (degrees)		
	x	y	z	Roll	Pitch	Yaw
Camera 1	-22.5	-425	215	-2.65	-28.3	-8.00
Camera 2	-6.28	-429	215	-2.18	-29.2	-5.61
Projector 1	226	-451	227	-1.74	-5.54	11.7
Projector 2	-94.4	-418	202	-2.97	-4.68	-24.4

Table 2. External parameters of the perspective devices in the room.

We verified the precision of the projector calibration by having a person stand at every corners of the the tiling in the room (61×61 cm) and by giving the algorithm the exact position on the floor. To account for the non-flatness of people, we used values for the width and height of the person that were respectively 110% and 103% of their real values. The maximum error we found on the floor was 54.3 cm, and the maximum error in z, 12.1 cm, while on average the errors were 28.0 cm and 2.0 cm respectively, all measurements precise within 2.0 cm.

(a) Without shadow removal. (b) With shadow removal.

Figure 7. Sample images of the room with one person standing, with and without shadow removal.

5.2. Tracking

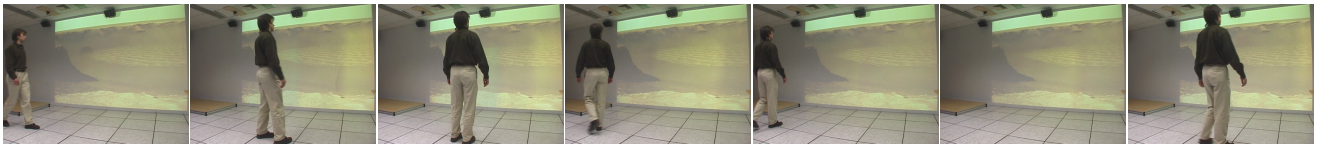
The focus of our work is not to compare different tracking algorithms, and we simply adopted the blob tracker from the OpenCV Video Surveillance Module. Although its performance was not tested thoroughly, it was obvious from our observations that it could not track with a precision greater than 10 centimeters. Reliability was also problematic, especially when a person occluded another person.

5.3. Shadow Removal

Typical results are shown in Figures 8 and 9, for a static image projection and a dynamic video projection, respectively. The full video is available at http://cim.mcgill.ca/%7Esaudet/research/procams2007.mp4 . Although the system currently achieves reasonable results in the case of static images, some additional refinement is necessary for it to perform well with video projection. In the case of static images, we could compensate for calibration and tracker imprecision by dilating the masks appropriately. Visible artefacts, such as the ones in Figures 8(a) and 8(b), are due mostly to fast movements not correctly predicted by the Kalman filter. However, in the case of video projection, the tracker frequently fails, resulting in large visible shadows such as seen in Figure 9(d).

An additional problem is that the system generates artefacts during occluder movement, such as in Figure 9(b). This is due, in part because it does not properly synchronize the mask display. Currently, each machine receives a mask, processes it, and sends it for display with no regards to whether other machines are ready for display. Obviously, a rudimentary synchronization technique would improve the results.

Since one of our objectives was to implement a system that could perform shadow removal in real time, we tested the performance in terms of processing delay. We benchmarked our current largely unoptimized codebase on an Intel Pentium 4 2.60 GHz. Applying the disparity map on 512×384 color images and tracking, using the resulting difference image resized down to 256×192, took



(a) frame 100 (b) frame 150 (c) frame 200 (d) frame 250 (e) frame 300 (f) frame 350 (g) frame 400

Figure 8. Frames from demo video with static image projection.

(a) frame 150 (b) frame 250 (c) frame 350 (d) frame 900 (e) frame 1100 (f) frame 1150 (g) frame 1200

Figure 9. Frames from demo video with dynamic video projection.

39.2 ± 3.4 (SD) ms. The generation of a 640×512 mask took 9.38 ± 0.94 (SD) ms, while mask post-processing (including a 50 pixel blurring) and correction took 79.1 ± 2.5 (SD) ms, for a grand total of 127.7 ± 6.8 (SD) ms. This does take into account delays in capture, display or network transfer.

6. Discussion

Compared to other methods for shadow removal, our object tracking method has several pros and cons. First, shadow detection [3, 6, 8, 13] requires that at least one camera be placed strategically to have an unoccluded view of the entire scene. In practice, this can sometimes be challenging. The infrared occlusion detection method [4, 17] requires one camera alongside each projector, in addition to the installation of near infrared floodlights at the display surface. In contrast, our method only requires that two cameras be placed appropriately for tracking people. Second, shadow detection methods cannot remove shadows before they appear. Although the simple approach of the infrared occlusion detection method, which preemptively dilates the occluded region to account for possible movement in any direction, appears effective at solving this problem in most cases, our method goes one step further and uses prediction information from a Kalman filter. Third, since both our method and infrared occlusion detection methods do not make use of projected images, it does not matter, in principle, whether we project static images or dynamic video. On the other hand, shadow detection becomes dramatically more complex in the case of video projection. Although Jaynes *et al*. [8] achieved nine frames per second, it is uncertain whether this can be easily increased to at least 24 frames per second. Flagg *et al*. [4] found that the combined latency of their camera and projector was in the order of 50 ms. For this reason, with commonly available hardware, the algorithm as developped by Jaynes *et al*. [8] cannot scale well beyond 15 frames per second.

Our method has drawbacks as well. The calibration required for both the shadow detection and infrared occlusion detections method can be relatively easily automated, whereas our approach currently requires a rather tedious calibration phase. Furthermore, tracking using cameras requires a sufficiently illuminated environment to make the occluders visible. Also, the simple tracking we used does not model limbs, and as such, effective shadow removal cannot be attained if the users wave their hands in front of the display. Moreover, tracking fails when a person occludes another person in the camera view. In this regard, shadow detection and infrared occlusion detection methods are superior. Fortunately, our approach is not bound to any particular tracker, so these limitations may be overcome by using better or simply more appropriate tracking methods.

In our implementation, we identified various sources of calibration and tracking errors. We found that large calibration errors may result because of improperly modeled non-linear distortion in the periphery of the camera image, uneven floor or wall, and the corrective homographies used for the projectors, which currently do not attempt to minimize geometric errors. We will investigate how to manage these errors, but considering all the issues related to calibration alone, we will also attempt to redesign the method to not require an elaborate calibration phase. In the case of the tracker, although it performs well for static images, it does not give acceptable results with video. Right now, it uses foreground information based solely on geometric properties of the cameras. However, we noted that, in the area of the display surface, even though the disparity map was precise, using it produced too much noise, probably cause by other intrinsic differences between the two cameras. We do not believe near infrared to be required, unless a dark environment is desired. A photometric color calibration should give us as good or even better performance with color cameras.

The work presented here provides an initial proof of con-

cept that shadow removal can be performed on today's hardware using conventional object tracking rather than requiring the more elaborate configuration of the near infrared occlusion detection approach. While there remains considerable effort ahead to refine and optimize our method, current results clearly demonstrate its feasibility. We hope that this will lead to better projector-camera systems now that we have shown that we can leverage object tracking technology already in place in many front projection environments, and achieve effective shadow removal.

Acknowledgements

This work was supported by a scholarship from the Natural Sciences and Engineering Research Council of Canada (NSERC). Picture from Mars, courtesy of NASA/JPL-Caltech.

References

[1] M. Ashdown and Y. Sato. Steerable Projector Calibration. In *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR '05) - Workshops*, volume 3, page 98. IEEE Computer Society, 2005.

[2] J.-Y. Bouguet. Camera Calibration Toolbox for Matlab, 2006. http://www.vision.caltech.edu/bouguetj/calib_doc/.

[3] T.-J. Cham, J. M. Rehg, R. Sukthankar, and G. Sukthankar. Shadow Elimination and Occluder Light Suppression for Multi-Projector Displays. In *2003 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR '03)*, volume 2, pages 513–520. IEEE Computer Society, 2003.

[4] M. Flagg, J. Summet, and J. M. Rehg. Improving the Speed of Virtual Rear Projection: A GPU-Centric Architecture. In *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR '05) - Workshops*, volume 3, page 105. IEEE Computer Society, 2005.

[5] R. I. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, second edition, 2004.

[6] M. N. Hilario and J. R. Cooperstock. Occlusion Detection for Front-Projected Interactive Displays. In *Advances in Pervasive Computing*. Austrian Computer Society (OCG), 2004.

[7] Y. Ivanov, A. Bobick, and J. Liu. Fast lighting independent background subtraction. *International Journal of Computer Vision*, 37(2):199–207, 2000.

[8] C. Jaynes, S. Webb, and R. M. Steele. Camera-Based Detection and Removal of Shadows from Interactive Multiprojector Displays. *IEEE Transactions on Visualization and Computer Graphics*, 10(3):290–301, 2004.

[9] P. KaewTraKulPong and R. Bowden. An Improved Adaptive Background Mixture Model for Realtime Tracking with Shadow Detection. In *2nd European Workshop on Advanced Video Based Surveillance Systems (AVBS01)*. Kluwer Academic Publishers, 2001.

[10] R. Raskar and P. Beardsley. A Self-Correcting Projector. In *2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR '01)*, volume 2, pages 504–508. IEEE Computer Society, 2001.

[11] R. Raskar, M. S. Brown, R. Yang, W.-C. Chen, G. Welch, H. Towles, B. Seales, and H. Fuchs. Multi-Projector Displays Using Camera-Based Registration. In *IEEE Conference on Visualization '99 (VIS '99)*, pages 161–168. IEEE Computer Society Press, 1999.

[12] R. Raskar, G. Welch, M. Cutts, A. Lake, L. Stesin, and H. Fuchs. The Office of the Future: A Unified Approach to Image-based Modeling and Spatially Immersive Displays. In *25th Conference on Computer Graphics and Interactive Techniques (SIGGRAPH '98)*, pages 179–188. ACM Press, 1998.

[13] R. Sukthankar, T.-J. Cham, and G. Sukthankar. Dynamic Shadow Elimination for Multi-Projector Displays. In *2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2001)*, volume 2, pages 151–157. IEEE Computer Society, 2001.

[14] J. Summet, G. D. Abowd, G. M. Corso, and J. M. Rehg. Virtual Rear Projection: Do Shadows Matter? In *CHI '05 extended abstracts on Human factors in computing systems*, pages 1997–2000. ACM Press, 2005.

[15] W. Sun. *Multi-camera object segmentation in dynamically textured scenes using disparity contours*. PhD thesis, Department of Electrical and Computer Engineering, McGill University, 2006.

[16] W. Sun and J. R. Cooperstock. An empirical evaluation of factors influencing camera calibration accuracy using three publicly available techniques. *Machine Vision Application*, 17(1):51–67, 2006.

[17] D. S. Tan and R. Pausch. Pre-emptive Shadows: Eliminating the Blinding Light from Projectors. In *CHI 2002 Conference on Human Factors in Computing Systems - Extended Abstracts*, pages 682–683. ACM Press, 2002.

[18] J.-P. Tardif, S. Roy, and M. Trudeau. Multi-projectors for arbitrary surfaces without explicit calibration nor reconstruction. In *Fourth International Conference on 3-D Digital Imaging and Modeling (3DIM '03)*, pages 217–224. IEEE Computer Society, 2003.

[19] B. Triggs. Autocalibration from Planar Scenes. In *ECCV '98: 5th European Conference on Computer Vision*, volume I, pages 89–105. Springer-Verlag, 1998.

[20] M. Wozniewski, Z. Settel, and J. R. Cooperstock. A framework for immersive spatial audio performance. In *The 2006 International Conference on New Interfaces for Musical Expression (NIME '06)*, pages 144–149. IRCAM - Centre Pompidou, 2006.

[21] Z. Zhang. A Flexible New Technique for Camera Calibration. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(11):1330–1334, 2000.