Machine Vision and Applications (2002) 13: 164–173

Machine Vision and Applications
© Springer-Verlag 2002

Head tracking using stereo

Daniel B. Russakoff and Martin Herman

National Institute of Standards and Technology, 100 Bureau Drive, Stop 8940, Gaithersburg, MD 20899, USA; e-mail: russakof@robotics.stanford.edu

Accepted: 13 August 2001

Abstract. Head tracking is an important primitive for smart environments and perceptual user interfaces where the poses and movements of body parts need to be determined. Most previous solutions to this problem are based on intensity images and, as a result, suffer from a host of problems including sensitivity to background clutter and lighting variations. Our approach avoids these pitfalls by using stereo depth data together with a simple human-torso model to create a head-tracking system that is both fast and robust. We use stereo data[1] to derive a depth model of the background that is then employed to provide accurate foreground segmentation. We then use directed local edge detectors on the foreground to find occluding edges that are used as features to fit to a torso model. Once we have the model parameters, the location and orientation of the head can be easily estimated. A useful side effect from using stereo data is the ability to track head movement through a room in three dimensions. Experimental results on real image sequences are given.

Key words: Security – Surveillance – Human motion – Human–computer interaction

1 Introduction

Human-head tracking has been an area of active research in computer vision for several years. The term *head tracking*, however, means different things to different people. Some think of it as the problem of figuring out not only the location of the head but also the 3D pose and sometimes even the complex facial expressions. Much research has been devoted to this difficult problem [26, 27, 28, 6, 29, 30, 31, 32, 33, 5]. We will, however, focus our efforts on a simplified version of this problem: the determination of 3D position and 2D orientation of the head in a sequence of images. Head position and orientation are important parameters for a variety of applications including virtual reality and telepresecence [1, 16, 17, 18, 19, 20, 21], augmented reality [22, 23], face recognition [2], voice recognition [24], audio equalization zone steering [25] and perceptual user interfaces [15].

2 Background

There has been very little work on head tracking using stereo. Until recently, systems that produce real-time, stereo depth data have been unavailable on the commercial market. As a result, most approaches to this problem have relied exclusively on intensity images must use color cues and intensity edges for face–head detection and tracking. In [4], they use color histograms and intensity gradients together with a second-order motion model and a local search to track skin-colored, elliptical face blobs. Similar methods are used by other researchers to track skin-colored blobs with various additions. In [34], they also add in a hypothesis-tree model to explicitly handle occlusions. [35] adds a blink detection module to augment the power of the color module, [36] uses a best-fit ellipse energy function to more accurately classify skin-colored regions as faces, [37] uses acoustical information to constrain the color information and [38] adds mouth-shape features to the color module. More recently systems that rely exclusively on color models have been pursued [39, 41, 40].

Not all methods track blobs of skin color, though. [42] and [15] track the contour of the head and shoulders using image intensity gradients. [43, 9, 44, 45, 7] use combinations of skin-color models, template models and various other cues to help detect and track heads. A less typical approach is the use of a variable numbers of wavelets to track intensity-based face templates [46].

All of these approaches, by virtue of their reliance on intensity and color information, are notoriously sensitive to environmental factors that affect intensity values such as changes in illumination or background clutter. Recent work in adaptive mixture modeling [48] may mitigate these effects somewhat, but they still cannot effectively handle self-shadowing and situations where the foreground and back-

Correspondence to: Daniel B. Russakoff

[1] Commercial equipment and materials are identified in order to adequately specify certain procedures. In no case does such identification imply recommendation or endorsement by the National Institute of Standards and Technology, nor does it imply that the materials or equipment identified are necessarily the best available for the purpose.

ground are similar. In addition, they cannot, in general, handle the full range of possible skin tones. Stereo depth calculations, however, do not encounter these problems. This observation, coupled with recent advances in stereo hardware that allow us to gather depth data in real time [10], suggests a new approach to the head-tracking problem.

There has been very little work done on head tracking using stereo. Some systems [11, 8] use stereo but still rely heavily on skin-tone pixel extraction, an intensity-based measure subject to all of the aforementioned problems. One approach [14] uses only stereo data, but its complicated models prevent tracking of rapid movements and so require the user to move slowly. In addition, this algorithm also requires a manual initialization step.

3 Basic idea and motivation

Our approach is to use stereo data to perform a more accurate foreground segmentation. We then use depth and intensity information to fit a simple torso model to the foreground, looking specifically for the occluding edges of the shoulders. Because the model is so simple, we can perform the fit to each frame separately without having to use traditional tracking techniques to limit the search space. This means that our tracker will not get confused as easily by rapid movements or temporary occlusions. Another important benefit of using stereo depth data is that, once we find the head in the depth image, if we know the cameras' focal lengths and baselines, we can easily determine its position in 3D world coordinates.

We hope to use this information in several ways. For one, this is an important complement to a smart room where knowing the 3D position of a user's head is a way to steer a microphone array to listen in that direction. We would also like to use it as a first step in the bootstrapping of a more complicated, articulated motion tracking system that can perform human-gesture recognition. In addition, work is underway to use this system to provide real-time, 3D head coordinates as an input to a face recognition algorithm similar to the one in [2].

The paper is organized as follows. Section 4 will discuss the algorithm for segmentation in detail. Section 5 will describe our torso model and its parameters, while Sect. 6 will discuss how we acquire that model in each frame. Section 7 presents our head localization algorithm. Our results and conclusions will be presented in Sects. 8 and 9, respectively.

4 Segmentation

Our system begins with a segmentation of the human figure in the foreground of our image sequence. Conventional approaches using intensity images [7] create a Gaussian model of the intensity over a certain interval of time of each pixel in the background and then determine whether a pixel is part of the foreground based on its distance from the background in the chosen color space. This method has two important limitations. First, it is extremely sensitive to variations in lighting conditions. For example, if the lighting suddenly changes, the background model is no longer valid and the resulting segmentation is incorrect. Second, the effects of shadows are very difficult to handle. If the foreground figure casts a shadow, the darkened region could differ enough from the background to be classified as foreground. In addition, if the foreground figure happens to be similar in color to the background, it will be classified as background.

4.1 Using depth data

The use of stereo eliminates the aforementioned problems. With depth images, we proceed as before, modeling each background pixel as a Gaussian with a mean μ and a standard deviation σ. This time, however, we build the model with depth instead of intensity values. Closely following the work of [12], once we build a depth model of the background, we can identify the foreground as any region where the depth is sufficiently closer to the camera than the background. This is much more physically intuitive than the intensity segmentation and more accurate as well. Because the nature of the stereo correlation calculation makes it insensitive to color, shadows, or lighting variations, we do not have to worry about the previous problems.

The segmentation, however, is not quite this simple. Stereo matching is extremely sensitive to image texture. In our case, the correlation-based stereo system has a great deal of difficulty operating in regions where there is little texture. For example, consider a blank wall. A stereo system attempting to correlate pixels in such an untextured region will have a difficult time finding the correct matches as all pixels look alike. The result is an area of incorrect matches yielding disparities more or less randomly distributed throughout a range dependent on the size of the correlation window. This noise, depicted in Fig. 1 is neither Gaussian nor white, making it very difficult to model.

Unfortunately, this adversely affects our segmentation as we cannot effectively model the background in regions without adequate texture. We can, however, identify those background pixels that are unreliable with a simple test of our model's standard deviation: if $\sigma_{ij} > \beta$ where β is a user-defined threshold (we used β = 2). To combat this problem in untextured regions, we devised our own segmentation scheme, closely related to [12] but with an important additional validation step.

4.2 Surface validation segmentation

Once we have modeled the background, we face the problem of picking out the foreground in a new disparity[2] image

[2] In the rest of the paper, we refer to disparity images obtained using Digiclops; these are the depth images used in our experiments. Digiclops is a commercial stereo system from Point Grey Research. It uses three cameras to do multiple-baseline stereo disparity calculation. Calibration is done in the factory and, according to Point Grey, the input images are rectified to fit an ideal stereo camera model within 0.06 pixels. The system also has a calibration retention system that makes it robust to shocks and vibrations. Noise is also an issue with these systems; however, the image transmission in Digiclops is completely digital (via an IEEE 1394 interface), which removes the problems of frame-grabber jitter and analog-to-digital conversion noise. That said, Digiclops suffers from the same limitations of every other stereo system: it does not perform well in areas without much

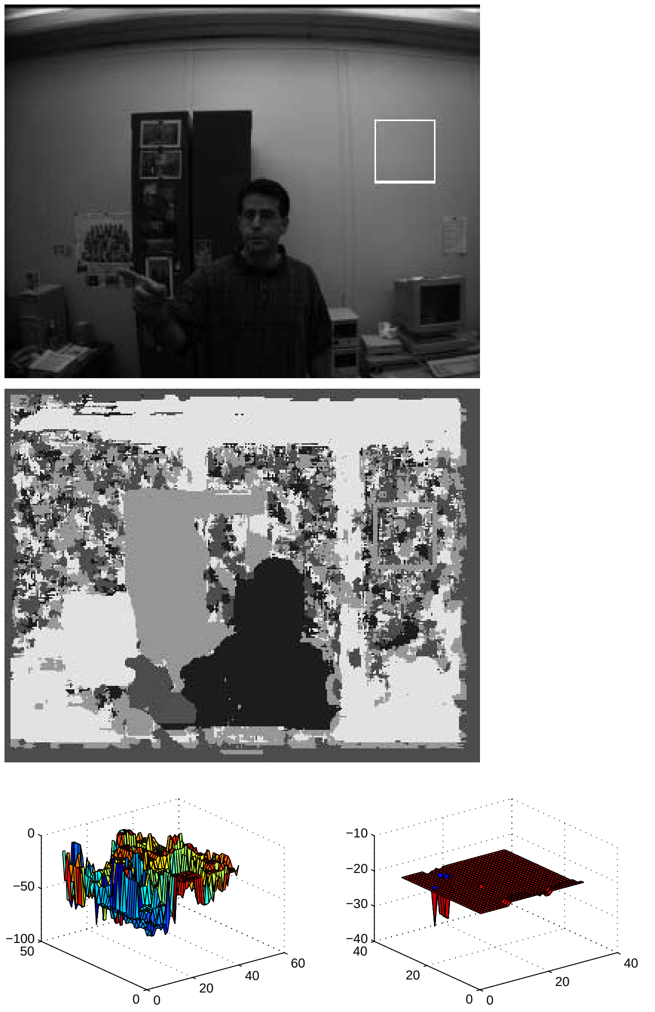

Fig. 1. *Top*: Sample input image from camera. *White square* highlights region with little texture. *Middle*: Disparity image from the Digiclops Stereo System from Point Grey Research, Inc. Note noise in the highlighted region. *Bottom left*: Reconstruction of physical surface based on disparity values in highlighted region (negative z axis trends away from camera). *Bottom right*: Reconstruction of physical surface based on region centered inside of the human figure in the middle

DI. In the case where a pixel of the foreground DI_{ij} is in front of a reliable background pixel ($\sigma_{ij} < \beta$), we have demonstrated that the segmentation is simple. All we must say is that a pixel is part of the foreground if the disparity value at that point is in front of the mean background by more than a standard deviation. When we are dealing with an unreliable background pixel ($\sigma_{ij} \geq \beta$), things get much more complicated. In these cases, since μ_{ij} is not a reliable representation of depth, we cannot know based only on the value of DI_{ij} whether that pixel is in front of the background or not. Here we make an important assumption: the foreground figure must consist of a smooth blob of pixels with similar disparity values. In other words, it should be distinguishable from untextured background in that its disparity values suggest a surface that is smooth and realistic like that in Fig. 1 bottom right, not noisy and spiked like the one in Fig. 1 bottom left. Assuming we can identify all regions in an image that can be considered smooth physical surfaces, segmentation of the foreground is as simple as identifying all such surface regions that occur in front of the background. To find these surfaces, we use a modified connected-components algorithm as follows:

1. Consider DI to be a graph G where every vertex G_{ij} corresponds to a pixel DI_{ij}.
2. For each vertex G_{ij}, connect it to its four neighbors if and only if $\sum_{n \in N_{ij}} |G_{ij} - G_n| < t$ where N_{ij} is the 4-neighborhood of vertex G_{ij} and t is a user-defined threshold. Since neighboring disparity values in untextured regions differ by large amounts, there is a great deal of latitude in choosing this threshold.
3. Connected components larger than a nominal size are accepted as surfaces and all other pixels are ignored. The size cutoff is also relatively easy to choose and is based on the assumption that the human figure will take up at least 10% of the image.

4.3 Segmentation algorithm

Thus, our segmentation algorithm is:

1. Using 20–30 images, model the background using a Gaussian $[\mu, \sigma]$ for each pixel.
2. Based on the values for standard deviation, determine unreliable background models by looking for pixels with $\sigma > \beta$ where β is a user-defined threshold (we used β = 2).
3. For each subsequent disparity image DI, calculate the areas considered to be physical surfaces.
4. A surface pixel DI^s_{ij} is classified as foreground if
   a) the background is reliable at $[i, j]$ and $DI^s_{ij} > \mu_{ij} + \sigma_{ij}$
   b) the background is unreliable at $[i, j]$
5. Finally, to eliminate all remaining noise, we run a binary connected-components algorithm and extract the largest component.

We have already looked at the first case in step 4, but the second deserves a bit of explanation. When we classify the background as unreliable, we are implicitly assuming that it is located in an untextured region. This is a safe assumption as it is generally only in those regions that the disparity value would fluctuate so much from frame to frame. As a result, the data in this region do not correspond to a physical surface and would not aggregate into a connected component large enough to classify as a surface. So, if we see a surface pixel where we expect to find an unreliable background pixel, we know that it must be part of the new foreground. Figure 2 illustrates these concepts.

5 Torso model

Once we have an accurate segmentation, we look to fit a simple torso model to the foreground figure. The model is based on the assumption that the figure is upright or leaning slightly to one side, a reasonable assumption for our domain

image texture, it does not handle specular reflections, and it cannot handle occlusions.

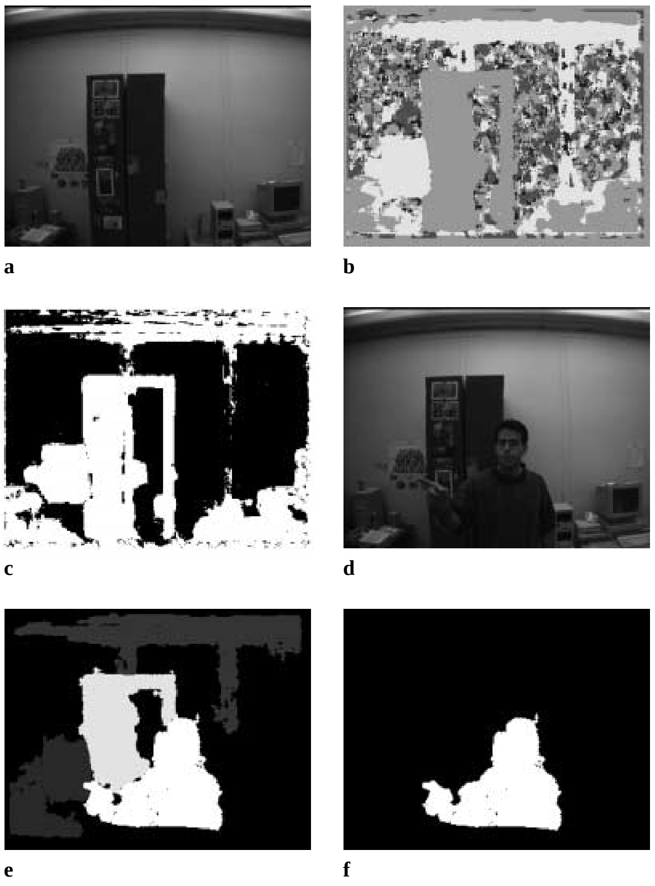

Fig. 2. Illustration of segmentation: **a** one of the images in the input sequence of background images; **b** disparity map output from Digiclops for this image; **c** illustration of reliable (*white*) background pixels based on the standard-deviation model (steps 1 and 2 of our algorithm); **d** new test image for foreground segmentation; **e** results from surface validation (step 3) with surfaces shown (all regions except black) passing the validation requirement; **f** final result of segmentation (steps 4 and 5)

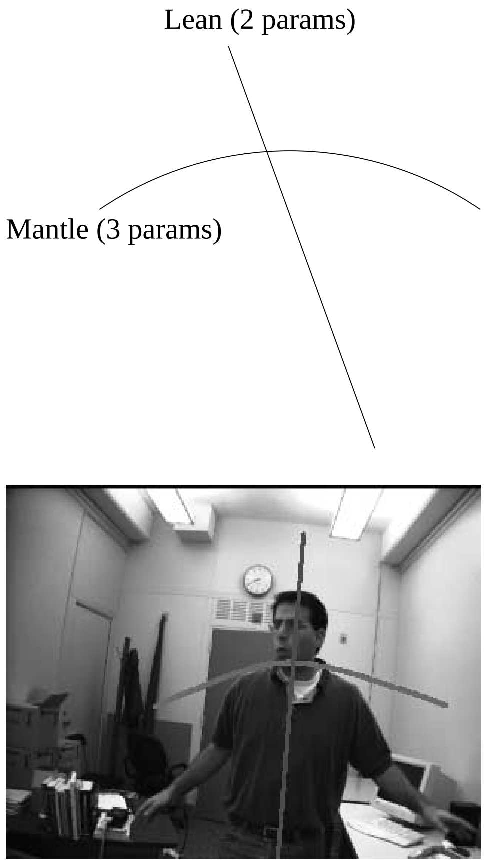

Fig. 3. *Top*: Illustration of simple torso model. *Bottom*: Application of torso model to image

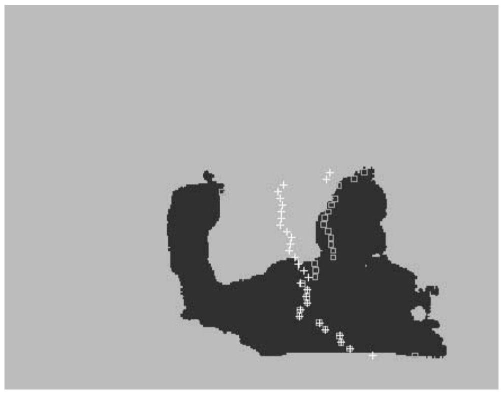

Fig. 4. Illustration of the horizontal mean values (*plus signs*) vs. horizontal median values (*squares*). Notice how much more the means are affected by the waving arm

of interest. Given this, we notice that the occluding edges of the shoulders, heretofore referred to as the *mantle*, are strong cues that vary very little with respect to the motion of an upright figure. Thus our torso model consists only of five parameters, two for the straight line that captures the general lean of the figure and three for the quadratic that traces the outline of the mantle. Figure 3 shows the model and its application to a real image. We loosely interpret the intersection of the lean and the mantle as being the neck point. This will be useful later to localize the head.

6 Model acquisition

Because we have such a simple model, it is relatively easy to acquire. Given the segmented foreground figure as a binary image, we take advantage of the depth information stereo imaging gives us to help us extract the lean in the following way:

1. For each row i of the binary image, calculate the median of the column values of the foreground pixels. We will call this value the *horizontal median*. We use medians instead of means because they are less sensitive to outliers in the foreground figure caused by waving arms (Fig. 4).
   In addition, we use a second level of outlier rejection based on the perceived 3D position of the figure. To do this, we need a rough approximation of the center (C_{fg}) of the figure in three dimensions. If we can make a reasonable guess about this point, we can ignore foreground pixels that are too far away horizontally from it (e.g. those belonging to waving arms). Since we are using stereo and have access to depth data, we can de-

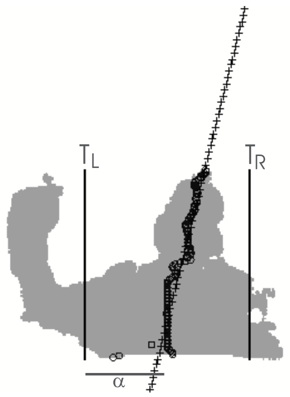

Fig. 5. Example of thresholds (depicted as *vertical bars*) in action. The *dark circles* represent the recalculated horizontal medians and the *plus signs* represent the line that best fits them

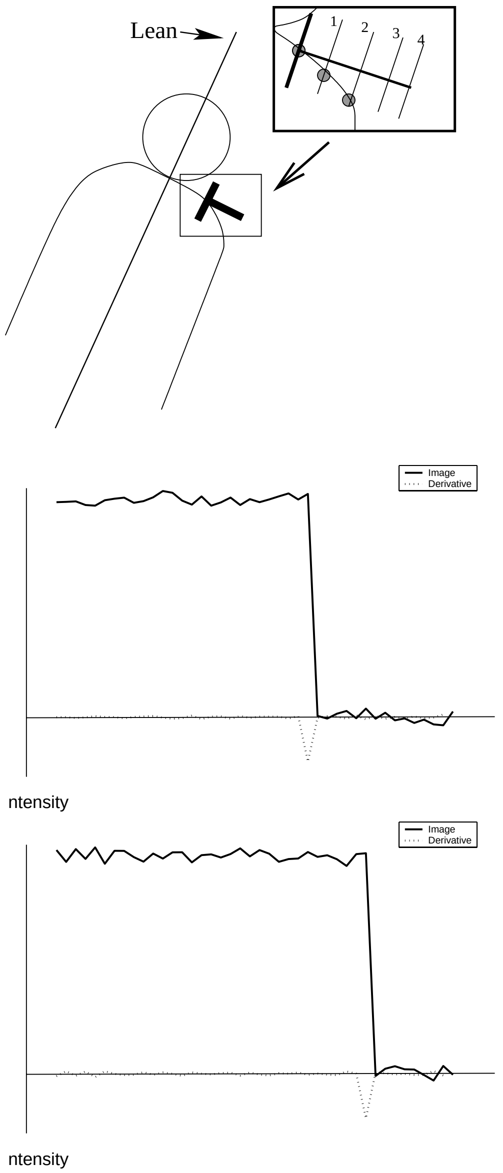

Fig. 6. *Top*: Example of a directed local edge detector. The detail of the figure shows examples of four of the slices along which the image gradient is calculated. The *gray dots* represent occluding edge points found by the detector. *Middle*: Graph of 1D slice of the image intensity together with its gradient along slice 1 of the edge detector above. *Bottom*: Same as above except with slice 2 of the edge detector

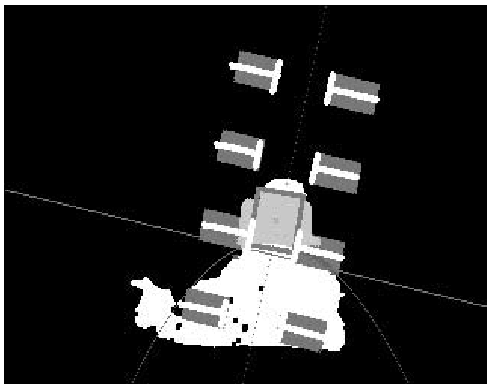

Fig. 7. This image shows the lean, the mantle and a few examples of the placement of the local edge detectors (*gray rectangles*). Also, the *light-grey area* above the mantle line represents points classified as being part of the head. The *x* represents the centroid (in image coordinates) of those head points

fine the term ‘too far away’ in world space and not in image space. This allows us to handle figures at any depth and maintain scale independence without having to resort to messy multi-resolution calculations. We make our guess about C_{fg} using the assumption that the majority of the lower area of the figure is usually evenly distributed around the center of the figure, a reasonable assumption for our domain of interest. This area encompasses a figure’s legs and lower- to mid-torso region, and its horizontal medians are very seldom disturbed by waving arms. We can get a value for the image coordinates of C_{fg} by finding the centroid of the horizontal medians calculated on the lowest 33% of the foreground figure. Once we have C_{fg} in image coordinates, we can use our depth data and the known camera parameters to project this point into 3D world coordinates (Fig. 5).

2. Perform outlier rejection in the following manner:
   – Set pixel threshold columns to the left (T_l) and right (T_r) of C_{fg} by calculating the distance in pixels represented by world displacements of α centimeters to the left and right of C_{fg}, respectively. The idea here is that α should approximate the half-width of a standard human figure
   – Using the thresholds, reject foreground pixel values as outliers if their column component does not fall within the left and right thresholds. Recalculate the horizontal medians based on this new information.
3. Using a singular value decomposition (SVD) line fit, find the best-fit line to the adjusted horizontal median values for each row. That line is the lean.

In our experiments, we used α = 25 cm, though there is a fair amount of latitude in this choice. We used SVD line fitting instead of other, less expensive approaches because of its numerical robustness. Specifically, its stability makes it a good deal less sensitive to perturbations of the data [49]. Figure 5 depicts an example of the lean acquisition.

Once we have the body lean, we can start acquisition of the mantle. As we mentioned before, the strongest cues are the occluding edges of the shoulders on either side of the head. To find these, we employ directed local edge detectors similar to those used in [13]. We orient these detectors

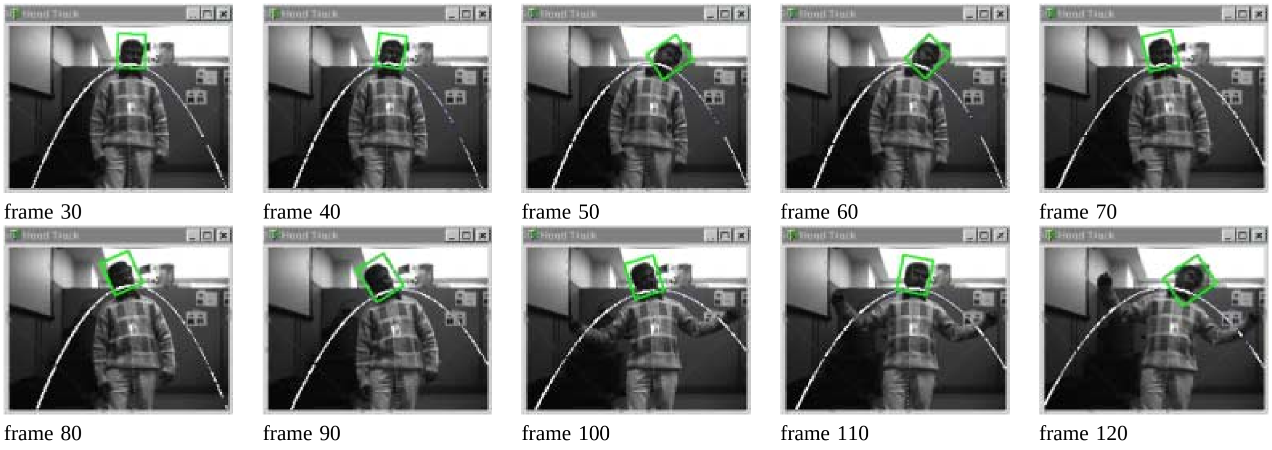

Fig. 8. Results from a sequence of rapid head movements. The images were acquired at about 2 Hz so this sequence lasts about 45 s

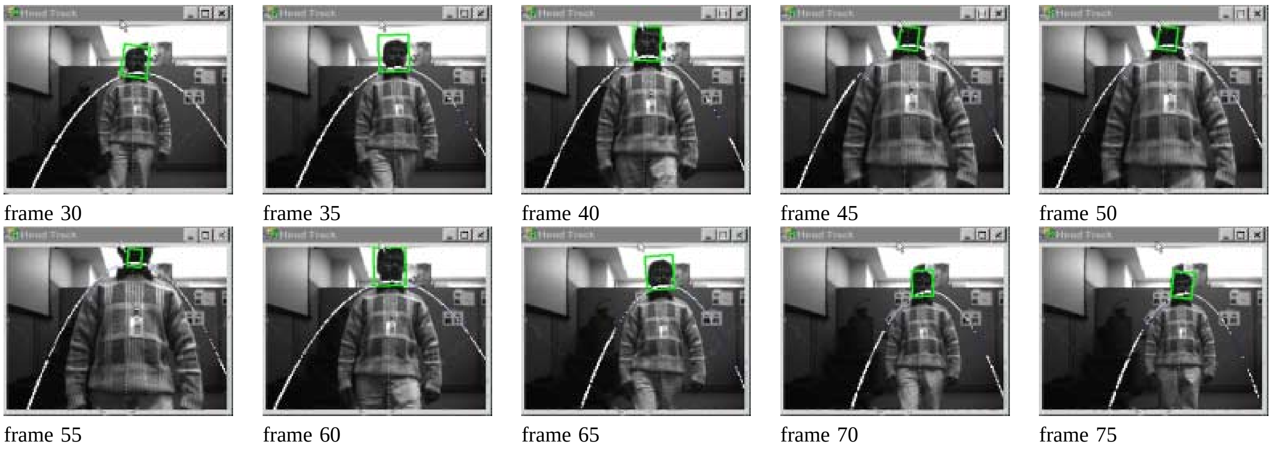

Fig. 9. Results from a figure approaching and then walking away from the camera. The images were acquired at about 2 Hz so this sequence lasts about 22 s

perpendicular to the lean and look for edges by thresholding the image gradient along slices perpendicular to the detector's orientation (parallel to the lean). Figure 6 offers an illustrated example of such an edge detector. To place these detectors in the best position, we use the depth data of the points along the lean to give us an estimate of how far the figure is from the camera. Based on this information, we can determine where, in image coordinates, to place the edge detectors. More specifically, once we decide where in world coordinates we would like to place the edge detectors, using our camera parameters we can project these points back into image coordinates. For example, if the figure is close to the camera, we are going to look for edges along a much longer line than if it is further away. Figure 7 shows an illustration of these concepts. We use the assumption that the typical head is 0.2 m wide and that the typical mantle is 0.4 m across.

We place a series of these local edge detectors up and down the image perpendicular to the body lean and keep a running tally of how many potential edge points we find. After searching the length of the body lean line, we select the pair of trackers that yield the most edge points and, using least squares, fit a quadratic to those points. That quadratic is the mantle and represents the final three parameters of our model.

7 Head localization

Once we have acquired a model, we calculate the intersection of the mantle and the lean, which we interpret as the neck. We then look radially out from the neck at points in the foreground that are:

– 'above' the mantle
– within a reasonable distance (0.2 m) in world coordinates from the neck (again, we can do this because we are working with 3D data)

After identifying such points, we calculate their centroid and make the assumption that, regardless of tilt, this point will represent the center of the head. We can now determine the orientation of the head simply by calculating the angle made by the line containing the head's centroid and the neck point. We assume that the distance between those two points is half of the height of the head and can easily draw a box around it (Fig. 7). Also, since we are using stereo, once we know the centroid of the head region, we can easily figure out its position in 3D.

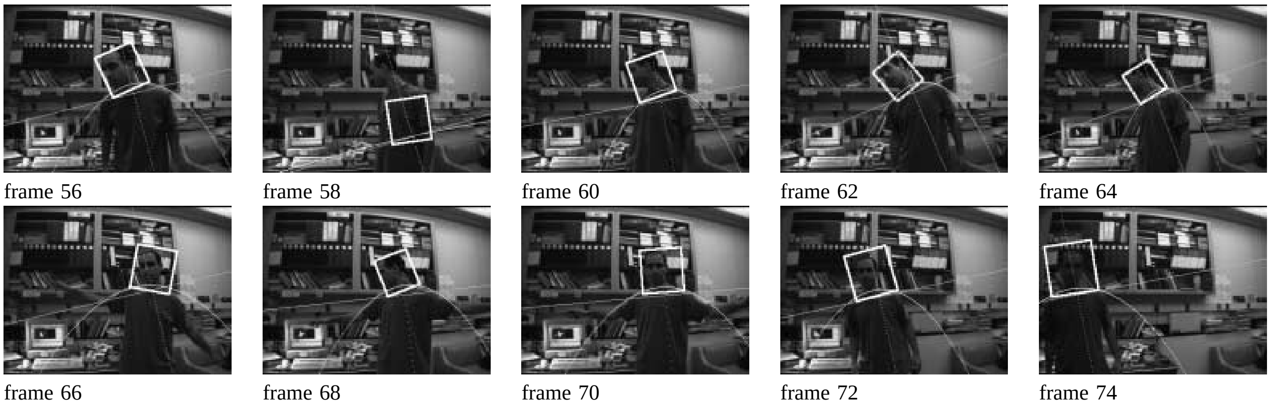

Fig. 10. Results from another rapid sequence of head and arm movements with a cluttered background. Notice the failure of the tracker (frame 58) when the figure is turned too much to the side reducing the strength of the shoulder cues. The timing is the same as in Fig. 9

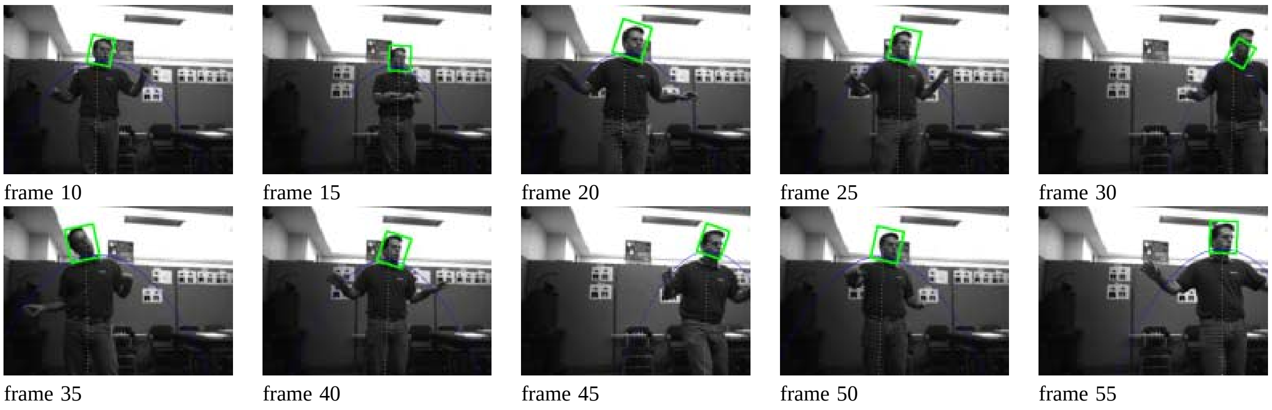

Fig. 11. More results from rapid sequence of head and arm movements with a different figure. The timing is same as in Fig. 9

8 Results

Figures 8–12 illustrate the results of our tracking scheme. They were all recorded using the same values for the aforementioned user-defined thresholds and parameters for each experiment. The choice of these parameters was easy for this particular domain and the experiments show that a single choice can handle different people and different motions in this domain. The sequences feature a variety of skin tones, cluttered backgrounds and rapid head movements that would be likely to confuse a tracker that relied on accurate predictions based on past motion. Plotted on each image is our acquired torso model as well as the orientation of the head. Figure 8 shows the tracker's ability to track changes in head orientation. Similarly, Fig. 9 illustrates the tracker's work on a figure approaching the camera and then moving away. Since we can adjust our algorithm using our knowledge of the depth of the figure, we maintain scale independence without any significant complications. Both figures also demonstrate our tracker's ability to work without any assumptions based on skin tone. The figure's dark skin is something that would confuse many of the trackers that rely on the identification of skin-colored pixels.

Figures 10 and 11 show the tracker's ability to work in the presence of waving arms and image clutter. Figure 10 also shows one of the failure modes of the system. When the assumption that the figure more or less faces the camera (a reasonable assumption for our domain of interest) is violated, the shoulder cues are not always strong enough to lead us to the correct configuration of the model. Fortunately, since our next step is entirely independent of the previous one, we are not confused for long and reacquire the figure soon after.

Figures 12 illustrates two failure modes of the tracker. At the top, the figure comes too close to the edge of the field of view so that our torso model cannot be acquired, and at the bottom, the figure's arms occlude the head and shoulders, obscuring our most important cues. In both cases, a simple tracker could easily get thrown off and have a difficult time finding the target again. In our case, however, regardless of where the figure is, we simply reacquire our torso model as soon as it becomes available again.

Figure 13 shows one of the important side effects from using stereo. Since we are using stereo and we know the cameras' intrinsic and extrinsic parameters, once we find where the head is located in image coordinates, we can easily turn that into a 3D point. As a result, we can track the movement of the head throughout a room in 3D.

8.1 Performance

This system is run using a resolution of 320×240 pixels and the processing time per frame is approximately 1 s on a dual 350 MHz Pentium II.

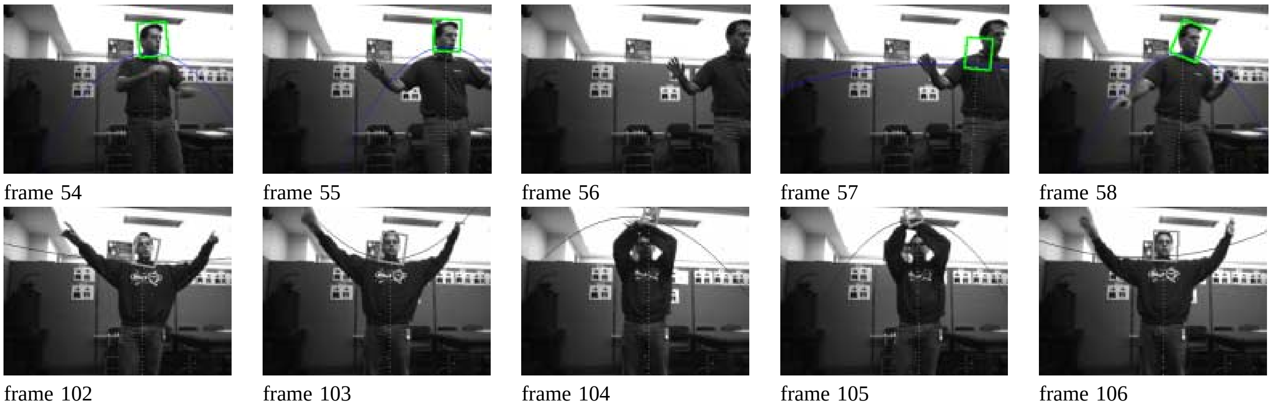

Fig. 12. *Top row*: Failure and reacquisition when figure moves out of field of view. *Bottom row*: Failure and reacquisition when figure's head and shoulders are occluded. The timing is the same as in Fig. 9

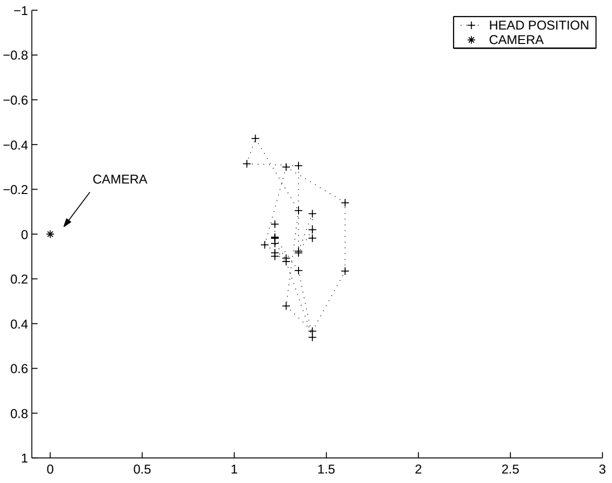

Fig. 13. Bird's eye view of head movement through room in sequence from Fig. 11. Both axes are in meters

9 Conclusion

What we have shown is a new approach to head tracking taking advantage of stereo depth data as well as the segmentation accuracy real-time stereo affords. We have created a simple torso model that is quick to acquire and does not require accurate predictions between frames to work. As a result, we can ignore the common assumption of small interframe motions as well as the problems generated by occlusions. We use this system to track heads in 3D throughout a room.

9.1 Future work

As mentioned earlier, work is underway to use the results of this algorithm as input to a steerable phased array of microphones in order to achieve more accurate voice recognition without the use of user-mounted microphones. Also, we hope to use the 3D position of the head as input to a face recognition algorithm or to bootstrap a more complicated articulated motion tracker.

As for extensions of the algorithm itself, a relatively easy one would be to track multiple heads in an image. This is a simple matter of identifying all of the blobs in the foreground and acquiring a torso model for each. Another extension under consideration is to add a very simple prediction step that would reduce the computation time to acquire a model but not sacrifice the robustness of our 'one image at a time' system. We could also integrate this prediction step as a separate module. In this way we might have the model acquisition and prediction act independently and then use a comparison function to decide which solution makes the most sense. This could potentially eliminate failure modes in which one of these approaches fails but the other does not.

Other interesting extensions could be used to handle more difficult failure modes. For example, if a figure is holding a large object that is occluding its shoulders, or if the figure is occluded by another person, our algorithm will fail. We would need to be able to either recognize that situation and handle it gracefully or, potentially, perform a more intelligent segmentation of the foreground into layers, recognize the boundaries between them and ignore all pixels except those actually belonging to the figure. This segmentation would obviously require extensive work, especially to make it recognize such complicated and smoothly varying boundaries in real time. However, advances in image segmentation techniques suggest that it might not be out of reach [47].

Acknowledgements. The authors would like the thank Michael Benharrosh and Vince Stanford for many fruitful discussions. Many thanks also go to Mani Muniyandi for his help with the results section.

References

1. Darrell T, Blumberg B, Daniel S, Rhodes B, Maes P, Pentland A (1995) Alive: dreams and illusions. In: ACM SIGGraph, Computer Graphics Visual Proceedings, July
2. Darrell T, Moghaddam B, Pentland A (1996) Active face tracking and pose estimation in an interactive room. In: IEEE Conference on Computer Vision and Pattern Recognition, San Francisco, 18–20 June
3. Birchfield S (1997) An elliptical head tracker. In: 31st Asilomar Conference on Signals, Systems, and Computers, Pacific Grove, Calif., 2–5 November, pp 1710–1714
4. Birchfield S (1998) Elliptical head tracking using intensity gradients and color histograms. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Santa Barbara, Calif., June, pp 232–237,
5. La Cascia M, Sclaroff S, Athitsos V (2000) Fast, reliable head tracking under varying illumination: an approach based on registration of texture-mapped 3D models. IEEE Trans Pattern Anal Mach Intell 22(4):322–336

6. Basu S, Essa I, Pentland A (1996) Motion regularization for model-based head tracking. In: Proceedings of the 13th International Conference on Pattern Recognition, Vienna, Austria, 25–30 August
7. Wren C, Azarbayejani A, Darrell T, Pentland A (1997) Pfinder: real-time tracking of the human body. IEEE Trans Pattern Anal Mach Intell 19:780–785
8. Azarbayejani A, Wren C, Pentland A (1996) Real-time 3-D tracking of the human body In: Proceedings of IMAGE'COM 96, Bordeaux, France, May
9. Niyogi S, Freeman W (1996) Example-based head tracking. In: IEEE 2nd International Conference on Automatic Face and Gesture Recognition, Killington, Vt., 13–16 October
10. Konolige K (1997) Small vision systems: hardware and implementation. In: Eighth International Symposium on Robotics Research, October
11. Darrell T, Gordon G, Woodfill J, Harville M (1998) Integrated person tracking using stereo, color, and pattern detection. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Santa Barbara, Calif., June, pp 601–609
12. Eveland C, Konolige K, Bolles RC (1998) Background modeling for segmentation of video-rate stereo sequences. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Santa Barbara, Calif., June
13. Rehg J, Kanade T (1994) Visual tracking of high DOF articulated structures: an application to human hand tracking. In: Proceedings of the European Conference on Computer Vision, May, pp 35–46
14. Jojic N, Turk M, Huang T (1999) Tracking self-occluded articulated objects in dense disparity maps. In: International Conference on Computer Vision, September, pp 123–130
15. Turk M (1996) Visual interaction with lifelike characters In: IEEE 2nd International Conference on Automatic Face and Gesture Recognition,Killington, Vt., 13–16 October
16. Essa I, Basu S, Darrell T, Pentland A (1996) Modeling, tracking and interactive animation of faces and head using input from video. In: Proceedings of Computer Animation, Geneva, June, pp 68–79
17. Saad EE, Caudell TP, Wunsch DC II (1999) Predictive head tracking for virtual reality. In: International Joint Conference on Neural Networks, Washington, July, pp 3922–3936
18. Sowizral HA, Deering MF (1999) The Java 3D API and virtual reality. IEEE Comput Graph Appl 19(3):12–15
19. Wegman E (2000) Affordable environments for 3D collaborative data visualization. Comput Sci Eng 2(6):68–72
20. Yanagida Y, Maeda T, Tachi S (2000) A method of constructing a telexistence visual system using fixed screens. In: Proceedings of the IEEE Virtual Reality 2000 Conference, New Bruswick, N.J., pp 117–124
21. Bimber O, Encamacao LM, Schmalstieg D (2000) Real mirrors reflecting virtual worlds. In: Proceedings of the IEEE Virtual Reality 2000 Conference, New Brunswick, N.J., pp 21–28
22. Kim D, Richards SW, Caudell TP (1997), An optical tracker for augmented reality and wearable computers. In: Proceedings of the IEEE 1997 Virtual Reality Annual International Symposium, Los Alamitos, Calif., pp 146–150
23. Feiner S, MacIntyre B, Hollerer T, Webster A (1997) A touring machine: prototyping 3D mobile augmented reality systems for exploring the urban environment. In: International Symposium on Wearable Computers, Cambridge, Mass., 13–14 October, pp 74–81
24. Mizoguchi H, Shigehara T, Yokoyama M, Mishima T (1998) Virtual wireless microphone: a novel application of real-time visual tracking and sound signal processing. In: Proceedings of the 37th SICE Annual Conference, July, pp 999–1004
25. Gardner WG (1997) Head tracked 3-D audio using loudspeakers. In: IEEE Workshop on Applications of Signal Processing to Audio and Acoustics, October, pp 19–22
26. Azarbayejani A, Starner T, Horowitz B, Pentland A (1993) Visually controlled graphics IEEE Trans Pattern Anal Mach Intell 15(6):602–605
27. Jebara TS, Pentland A (1997) Parametrized structure from motion for 3D adaptative feedback tracking of faces. In: IEEE Conference on Computer Vision and Pattern Recognition, San Juan, Puerto Rico
28. Horprasert T, Yacoob Y, Davis LS (1996) Computing 3-D head orientation from a monocular image sequence. In: Proceedings of the 2nd International Conference on Face and Gesture Recognition, Killington, Vt., 13–16 October
29. DeCarlo D, Metaxas D (1996) The integration of optical flow and deformable models with applications to human face shape and motion estimation. In: IEEE Conference on Computer Vision and Pattern Recognition
30. Li H, Rovainen P, Forcheimer R (1993) 3-D motion estimation in model-based facial image coding. IEEE Trans Pattern Anal Mach Intell 15(6):545–555
31. Terzopoulos D, Waters L (1993) Analysis and synthesis of facial image sequences using physical and anatomical models. IEEE Trans Pattern Anal Mach Intell 15(6):569–579
32. Essa IA, Pentland AP (1997) Coding analysis, interpretation, and recognition of facial expressions. IEEE Trans Pattern Anal Mach Intell 19(7):757–763
33. Black MJ, Yacoob Y (1997) Recognizing facial expressions in image sequences using local parameterized model of image motion. Int J Comput Vision 25(1):23–48
34. Fieguth P, Terzopoulos D (1997) Color-based tracking of heads and other mobile objects at video frame rates. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp 21–27
35. Crowley J, Berard F (1997) Multi-modal tracking of faces for video communication. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp 640–645
36. Sobottka K, Pitas I (1996) Segmentation and tracking of faces in color images. In: Proceedings of the International Conference on Automatic Face and Gesture Recognition, pp 236–241
37. Collobert M, Feraud R, Le Tourneur G, Bernier O, Viallet JE, Mahieux Y, Collobert D (1996) LISTEN: a system for locating and tracking individual speakers. In: Proceedings of the International Conference on Automatic Face and Gesture Recognition, pp 283–288
38. Oliver N, Pentland A, Berard F (1997) LAFTER: lips and face real-time tracker. In: IEEE Conference on Computer Vision and Pattern Recognition, pp 123–130
39. Jordao L, Perrone M, Costeira J, Santos-Victor J (1999) Active face and feature tracking. In: Proceedings of the International Conference on Image Analysis and Processing, pp 572–576
40. Yachi K, Wada T, Matsuyama T (2000) Human head tracking using adaptive appearance models with a fixed-viewpoint pan-tilt-zoom camera. In: Proceedings of the International Conference on Automatic Face and Gesture Recognition, pp 150–155
41. Qian RJ, Sezan MI, Matthews KE (1998) A robust real-time face tracking algorithm. In: Proceedings of the International Conference on Image Processing, pp 131–135
42. Isard M, Blake A (1996) Contour tracking by stochastic propagation of conditional density. In: Proceedings of the European Conference on Computer Vision, pp 343–356
43. Raja Y, McKenna SJ, Gong S (1998) Tracking and segmenting people in varying lighting conditions using colour. In: Proceedings of the International Conference on Automatic Face and Gesture Recognition, pp 228–233
44. Triesch J, von der Malsburg C (2000) Self-organized integration of adaptive visual cues for face tracking. In: Proceedings of the International Conference on Automatic Face and Gesture Recognition, pp 128–134
45. Huang FJ, Chen T (2000) Tracking of multiple faces for human–computer interfaces and virtual environments In: IEEE International Conference on Multimedia and Expo, pp 1563–1566
46. Kruger V, Sommer G (1999) Affine real-time face tracking using a wavelet network In: International Conference on Computer Vision, pp 141–148
47. Felzenszwalb P, Huttenlocher D (1998) Image segmentation using local variation In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Santa Barbara, Calif., June, pp 98–104
48. Stauffer C, Grimson WEL (1999) Adaptive background mixture models for real-time tracking In: IEEE Conference on Computer Vision and Pattern Recognition, pp 246–252, November
49. Heath M (1997) Scientific computing: an introductory survey. McGraw-Hill, New York

Daniel Russakoff received an AB in geophysics from Harvard University in 1996 and an MS in computer science from Stanford University in 1999. He spent a year working in the Smartspaces Laboratory at the National Institute of Standards and Technology before returning to Stanford to pursue a PhD in computer science. His research interests include stereo, tracking, and medical image registration.

Martin Herman received a PhD in computer science from the University of Maryland. He is currently chief of the Information Access Division, National Institute of Standards and Technology (NIST). He is responsible for the overall program in research, measurements, testing, and standards in information access technologies at NIST, including speech processing and human language technology, multimedia information retrieval, image recognition, visualization and usability testing, and smart spaces. Previously, he was group leader of the Perception Systems Group at NIST, and held a faculty appointment at Carnegie Mellon University. He has performed research in computer vision, smart spaces, robotics and automated manufacturing, and has published over 75 papers in these areas. He was program co-chair of the IEEE Workshop on Applications of Computer Vision, 1992, and was a guest associate editor for the IEEE Transactions on Systems, Man, and Cybernetics' special issue on unmanned vehicles and intelligent robotic systems, 1990.