FREIE UNIVERSITÄT BERLIN

High Resolution Segmentation with a Time-of-Flight 3D-Camera using the Example of a Lecture Scene

Neven Santrac, Gerald Friedland, Raul Rojas

B-06-09
September 2006

FACHBEREICH MATHEMATIK UND INFORMATIK
SERIE B · INFORMATIK

High Resolution Segmentation with a Time-of-Flight 3D-Camera using the Example of a Lecture Scene

Neven Santrac, Gerald Friedland, Raúl Rojas
Freie Universität Berlin
Department of Mathematics and Computer Science
Takustr. 9, 14195 Berlin, Germany
{santrac, fland, rojas}@inf.fu-berlin.de

Abstract

Time-of-flight principle range cameras are becoming more and more available. They promise to make the 3D reconstruction of scenes easier, avoiding the practical issues resulting from 3D imaging techniques based on triangulation or disparity estimation. Their low resolution and frame rates as well as their low signal-to-noise ratio, however, kept many researchers from seriously considering this technology for segmentation purposes. The following article uses a practical application as an example to present both, an analysis of the problems resulting from the use of time-of-flight principle 3D cameras and a solution to deal with these issues. A lecturer is extracted out of a video that shows him in front of an electronic chalkboard. The extracted instructor can later be laid over the board semi-transparently allowing the students to look through him in a later video replay. Using the 3D Time-of-Flight camera technology improves the robustness of the segmentation and lowers its computational complexity.

1 Introduction

The underlying scenario presented in this paper is deduced from the practical experiences of using our e-learning environment called E-Chalk [21]. When using E-Chalk, a lecturer stands in front of a classroom writing on an electronic chalkboard. The electronic chalkboard captures the instructor's handwriting and stores them as vector data. The system records the creation of the board content along with the voice of the instructor to allow for live transmission and archival of lectures. With an electronic board to capturing strokes instead of using a camera to film the blackboard, the drawings can be rendered as a crisp image with the usual video compression artifacts that cause a disturbing blurring of the edges. However, when only the board image is transmitted mimics and gestures of the instructor are lost and the replay sometimes appears quite unnatural when drawings appear from the void. For this reason, many lecture recording systems do not only transmit the slides or the board content but also an additional video of the instructor. However the issue of split attention arises because we have two areas of the screen competing for the viewer's eye: the video window showing the instructor and the board window (for a detailed explanation of this psychological issue see [18]). In order to solve this problem, the idea is to create an additional film of the instructor showing him or her in front of the board and then extracting his or her image from the video. The image of the lecturer must then be separated in real-time from the steadily changing board background. Having performed this task, the image of the instructor can then be overlaid semi-transparently on the board, creating the impression that the lecturer is working directly on the screen of the remote student. Mimics and gestures of the instructor appear in direct correspondence to the board content (see Figure 1) which helps the student to better asso-

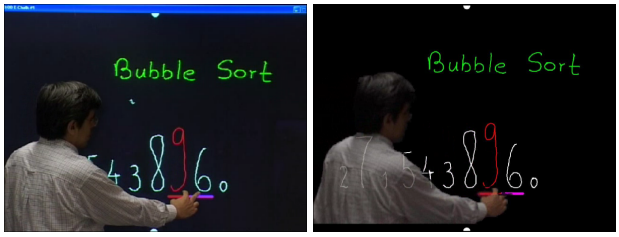

Figure 1. The left picture shows the scene as acquired by the 2D-camera. The data written on the electronic chalkboard is stored as vector data. The goal is to extract the teacher out of the scene and overlay his image on the vector data for replay. The result is shown in the right picture.

ciate the lecturer's gestures with the board content. Furthermore, no board content is occluded by the lecturer even if he or she is standing right in front of it.

In order to achieve this effect, a robust pixel-accurate segmentation technique has to be used that is feasible for regular operation in lecture rooms. This document presents that an appropriate solution for this task can be built using a combination of a 3D time-of-flight camera plus a normal video camera. Both cameras capture the instructor's 3D image and a corresponding video resolution 2D image respectively. Although our experiments focus on solving this particular segmentation problem the results can be generalized to various other object segmentation tasks.

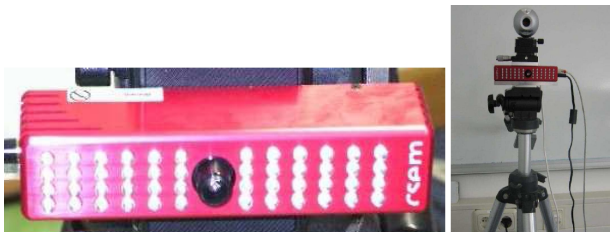

Figure 2. The Swissranger SR-2 3D-camera (left picture) and the 2D/3D-camera combination mounted on the self-developed pod (right picture).

2 Related Work

The standard technology for overlaying foreground objects onto a given background is chroma keying [11]. This technique is not applicable for many segmentation problems because the background of a scene is rarely fixed or monochromatic.

Separating the foreground from either static or dynamic background is the object of current research, see for example [16]. Much work has been done on tracking objects for computer vision (like robotic soccer [23], surveillance tasks [15], or traffic applications [2]), and also on interactive segmentation for image processing applications [3]. Numerous computationally intensive segmentation algorithms are being developed in the MPEG community, for example [5]. In computer vision, real-time performance is more relevant than segmentation accuracy as long as the important features can be extracted from each video frame. For photo editing applications, accuracy is more important and algorithms can rely on information obtained through user interaction (see discussion in [22]). For the task we investigate here, the segmentation should be as accurate as possible and non-interactive. A real-time solution is needed for live transmission of lectures.

The use of stereo cameras for the reconstruction of depth information has been thoroughly investigated. Disparity estimation is a calculation intensive task. Since it involves texture matching, it is affected by the same problems as texture classification methods, that is, similar or homogeneous areas are very difficult to distinguish and real-time processing requires additional hardware [25]. However, [14] already showed, that range information can principally be used to get a better sample of the background faster.

Weingarten, Gruener and Siegwart [24] describe the use of a 3D-camera for robot navigation. They use a differential-driven robot equipped with a time-of-flight 3D-camera in front of two horizontal laser range scanners and some ultrasound and infrared distance sensors. Although their article contains useful information about the working principles and capabilities of 3D time-of-flight cameras it does not concern video segmentation.

Göktürk and Tomasi [13] investigated the use of 3D time-of-flight sensors for head tracking. They used the output of the 3D camera as input for various clustering techniques in order to obtain a robust head tracker. This article investigates lecturer segmentation using time-of-flight principle 3D cameras. This is different because instead of only a head position, a pixel accurate boundary of the instructor has to be found in each frame.

Diebel and Thrun [7] use Markov Random Fields to increase the resolution of 3D cameras. They also use a combination of a 2D Camera and a 3D camera. They exploit the fact that discontinuities in range (i.e. in the 3D image) and coloring (i.e., in the 2D image) tend to co-align. We also experimented with their approach. The experiments showed that it works reasonably well. However, the method is computationally expensive and far from real-time performance. A downside is that the method sometimes blurs the edges between instructor and chalkboard. Sometimes the influence of the 2D camera image gets too strong and reflections from the board surface appear in the final segmentation result.

3 Hardware

3.1 The Time-of-Flight Principle

Time-of-flight principle 3D cameras are becoming more and more available (see for example [6, 20, 4, 1]) and their acquisition costs are continually decreasing. The experiments described herein were conducted using a miniature camera called SwissRanger SR-2 [6] built by the Swiss company CSEM.

Time-of-flight principle cameras work similar to radars. The camera consists of an amplitude-modulated infrared light source and a sensor field that measures the intensity of backscattered infrared light. The infrared source is con-

stantly emitting light that varies sinusoidal between a maximum and a minimum. Objects of different distances to the camera reflect the light in different intensities. The reason is that, at the same moment, objects that have different camera distances are reached by different parts of the sinus wave. The incoming reflected light is then compared to the sinusoidal reference signal which triggers the outgoing infrared light. The phase shift of the outgoing versus the incoming sinus wave is then proportional to the time of flight of the light reflected by a distant object. This means, by measuring the intensity of the incoming light, the phase-shift can be calculated and the cameras are able to determine the distance of a remote object that reflects infrared light. The output of the cameras consists of depth images and a conventional low-resolution gray scale video, as a byproduct. A detailed description of the time-of-flight principle can be found in [17, 19, 12].

The depth resolution depends on the modulation frequency. For our experiments we used a frequency of 20 MHz which gives a depth range between 0.5 m and 7.5 m, with a theoretical accuracy of about 1 cm. Usually, time-of-flight cameras allow to configure frame rate, integration time, and a user-defined region of interest (ROI) by writing to internal registers of the camera. The cameras then calculate the distances in an internal processor. The resolution of the SwissRanger camera is 160×124 non-square pixels. Unfortunately, 3D time-of-flight cameras are not yet available in higher resolutions such as NTSC or PAL.

3.2 2D/3D-camera combination

In order to capture a high resolution 2D color image, a standard web-camera from Logitech with a resolution of 640×480 is mounted on the time-of-flight camera using the self-developed stand shown in Figure 2. The idea is to correlate the high resolution 2D image with the low resolution 3D range data using this combination of 3D and 2D-camera. Both cameras are configured such that they see the same image. If the 3D-camera had the same resolution, the same lens-distortion, and its lens laid at the same position as the 2D-camera's lens, the range image and the 2D color image would directly correlate. The segmentation problem described here would then reduce to a simple depth range check and a cut-out of all those pixels that are closer to the 3D-camera than the board.

Unfortunately, the 2D camera image and the range image do not correlate directly. The reason for this is that the lens and image characteristics of the two cameras differ significantly (compare Figure 3). These must be corrected by software.

4 Technical Issues

4.1 Motion blur

The time-of-flight principle assumes that an object does not move during a measurement period. Objects that move too fast appear blurred. This makes finding an accurate boundary between the moving object and the rest of the scene difficult. In our example scenario, this poses a problem with the teacher's hands because they move fast while writing.

4.2 Light scattering

The 3D-camera illuminates the scene actively. However, the objects hit by the light reflect it not only towards the camera but in all directions possible. Objects that are close to the camera (less than about 75 cm) reflect too much light thus obstructing the correct depth measurement by an over saturation of the light sensors.

In our scenario, the working depth of the camera is between 50 cm and 250 cm. As soon as the teacher steps in front of the board, he or she also reflects some of the light onto the board, thus creating a silhouette around him. This makes the board appear closer than it actually is. The optimal depth for the 3D-camera would be between 2 m and 4 m but this would make the actual region of interest smaller in the depth image and make the effective resolution of the range image even smaller.

Figure 3. The left picture shows the depth data of the scene that is shown in the right picture. The pictures where acquired with the 2D/3D-Camera combination.

4.3 Camera synchronization

A major issue is the camera synchronization. The two cameras have different maximal frame rates and although the 3D-camera could theoretically work with a higher frame rate, practically, this does not come into consideration because the exposure-time would have to be lowered (thus making only very near objects measurable). The 3D-cameras do not yet support triggering by device drivers, so

a precise synchronization of the two cameras is not easily possible.

By setting the frame rates of both cameras to 10 fps and acquiring the data periodically, we are able to find out how many milliseconds need to pass after retrieving the data from one camera and before retrieving the data from the other camera. Even after this soft-syncronisation there is still a slight offset between the frames. In the worst case, we observed a spatial offset of up to 50 border pixels in the 2D image. For fast movements, the average error is about 10-30 pixels.

4.4 Noise

During our experiments we noticed that there is a substantial amount of noise from reflecting surfaces such as monitor screens. Experiments in front of a non-reflecting, white wall showed that the amount of noise depends on the reflection properties of the illuminated object. The noise from reflecting surfaces in the depth image also correlates with the image position: While in the center of the scene noise is not a major issue, the signal-to noise ratio shrinks radially from the center of the captured depth image. The error caused by noise makes up to 30 cm in the outer parts of the image. Of course, this reduces the precision of the 3D-camera seriously. In our scenario, it is nearly impossible to distinguish the teacher from the board if he or she comes too near. The hands are especially hard to separate because they almost touch the board.

4.5 Lens distortion

The two cameras have different lens distortions. The 3D-camera's lens is wider and there is also a stronger radial distortion. Practically, the two lenses can of course not be put at the same position. This causes an offset between the two images of the scene. Furthermore there is a depth-distortion (probably caused by different lens sizes), so if an object moves away from the camera, it will "shrink" in one of the cameras faster than in the other one.

4.6 Different resolutions

Because the resolution of the 3D-camera is only 160×124 the calibration (as explained in the next Section) basically scales the picture up to 640×480 pixels and transforms the pixel positions in respect to the offset of the different lens distortions. The gaps between the transformed pixels of the 3D-camera scene are filled by bilinear interpolation.

Figure 4. Without any calibration, segmentation by depth interval checking would look like this.

5 Calibration

As explained in the last Section, the images of the same scene as acquired by the two cameras differ (compare Figure 4). In order to calculate the mapping between the two images, a calibration step has to be performed prior to using the camera combination for segmentation tasks.

If we scale the 3d-camera's image to 640×480 we are able compare them directly. There is an offset caused by the different positions of the cameras that can be fixed by a simple translation. Now, the two images are centered around the same point. However, with growing distance from the center, the points are bent more rapidly in the image of the 3D-camera than in the image of the 2D-camera because of the different radial distortions.

So the difference between the same world points as acquired by the two cameras can be described by a polynomial, one for each dimension (x, y). Coefficient 0 is the offset between the lenses, coefficient 1 is a linear scaling factor, and the second, third, or higher coefficients represent any curvatures, or other mappings.

We measure the position of characteristic points in the worlds of the 3D-camera and the 2D-camera, compute their difference, and perform a curve fitting. We use a calibration pattern with at least 5×5 characteristic points, distributed as evenly as possible on the screen of both cameras (see Figure 5).

Any pattern (for example, circles, squares, ellipses, or line intersections) can be used as calibration template as long as it is roughly evenly distributed along the screen of both cameras and we can see the difference between the points in every part of the screen of the two cameras. For automatic software recognition of the points we used big black circles printed on a sheet of paper. We also tried to project a pattern using an overhead-projector, but this was invisible to the 3D-camera as it only senses reflected light from real objects - such as pigments on a sheet of paper.

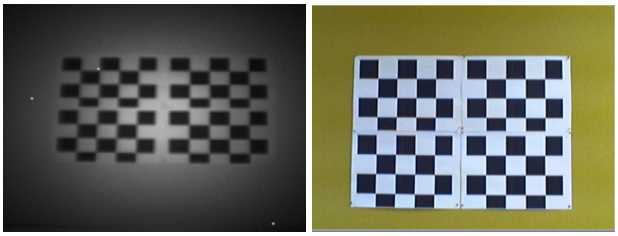

Figure 5. Left: The scene as captured by the 3D-camera (an amplitude image) Right: The scene as captured by the 2D-camera. The difference between the two images is noticeable. This pattern is used for calibration.

5.1 Formal Description of the Calibration

Let S_{2D} be the scene as acquired by the 2D-camera and S_{3D} be the scene as acquired by the 3D-camera (originally a 160×124 pixel image scaled to 640×480 pixels using a bilinear filter). Let n denote the number of the characteristic world points, p_i the characteristic world points as captured by the 2D-camera and P_i those captured by the 3D-camera:

$$\vec{p_i}, \vec{P_i} \in [0..639, 0..479], i = 0, .., n-1$$

We look for a function $F : R^2 \rightarrow R^2$:

$$F(x_3, y_3) = (x_2, y_2)$$

$$(x_3, y_3) \in S_{3D}, (x_2, y_2) \in S_{2D}$$

We can divide F into two functions - one for each axis:

$$F(x_3, y_3) = (F_x(x_3), F_y(y_3)) = (x_2, y_2)$$

$$F_x : R \rightarrow R, F_y : R \rightarrow R$$

Using the acquired point pairs (p_i, P_i) we can approximate F_x, F_y. We use steepest descent with the following error function (analog for y-axis):

$$E_x := \sqrt{\sum_{i=0}^{n-1} (F_x(x_{3_i}) - x_{2_i})^2}$$

$$x_{3_i} \in (p_i)_x, x_{2_i} \in (P_i)_x$$

As function F_x we use a polynomial (analog for F_y):

$$F_x(x) = a_0 + a_1 x + .. + a_k x^k$$

Since we use steepest descent, we need to compute the gradient of E with respect to F. More precisely, we deviate to each coefficient a_i of F and update the steepest descent iteration rule. We stop when the error is small enough. Experiments showed that polynomials of third degree were sufficient for a good approximation.

However, depth-distortion is still another problem. The calibration as presented above works only in one depth. In order to get an appropriate depth calibration, the calibration step is repeated in different depths. A linear interpolation between the polynomials for two nearest depths finally provides the entire mapping.

There is still a residual error because we assumed that all characteristic world points have the same distance to the camera. Although they are on the same plane, they do not have the same distance for the camera's lens. The closest point is in the middle of the lens, radially progressing farther away. Although this error proved to be ignorable it is easy to adjust the transformation function F to correct this theoretical issue.

$$F : R^3 \rightarrow R^2$$

$$F(x_3, y_3, z_3) := (A(x_3)+C(z_3), B(y_3)+D(z_3)) = (x_2, y_2)$$

A, B, C, D are polynomials, A and B have the same function as F_x and F_y, C and D are the depth dependent offsets.

The result is that only four polynomials need to be computed for all depths (through experimentation: A, B have degree 3, C, D degree 1).

Figure 6. Noise introduces a depth-measurement error of up to 30 cm. Which makes a segmentation exclusively by depth-range checking impossible.

Theoretically, a simple depth-interval check, every pixel between depths A and B is ignored, can be used to segment the 2D image after the two camera outputs have been correlated. In practice, however, this proved to be insufficient because of the low signal-to-noise ratio caused by light scattering (compare Figure 6). Furthermore the desynchronization of the cameras create a visible distortion (compare Figure 7). At first, we tried to minimize the noise influence by the use of standard image processing operations, such as gauss filters, media filters, or dilate/erode.

Figure 7. Left: The scene as captured by the 2D-camera. Right: The segmentation by depth range check after calibration. The extra pixel layer around the teacher is actually two or three pixels wide. The table is also recognized as foreground because it lies in the same depth interval.

Although this helps for an empty scene, light-scattering gets worse as soon as the instructors comes into the picture. The body of the person reflects light on the board area surrounding him or her. In the result, segmentation by depth range check alone is not precise enough. However, using the depth range check it is easy to extract a superset of the instructor robustly. This makes it possible to combine the output with software segmentation methods.

6 Segmentation using SIOX

In order to get a pixel-accurate segmentation result we feed the output acquired as described in the previous sections in the SIOX engine [8, 9]. SIOX stands for "Simple Interactive Object Extraction" and is a generic image and video segmentation engine.

SIOX segments a given image into foreground and background based on the color characteristics of the two classes. The input for the algorithm consist of an image or video frame and a set of characteristic colors for foreground and background. The scene is then segmented with sub-pixel accuracy [10].

The basic idea is to use the 3D-camera to collect the background and foreground color samples for SIOX. Using the depth data, we are able identify several areas that can very robustly classified as foreground and background. Since only a few characteristic colors suffice, we can safely ignore many of the errors described above including the blurring of the lecturer's border pixels caused by light-scattering, camera de-synchronization, noise, etc.

6.1 Acquiring representative colors

A good subset of the teacher can be easily found by removing border pixels with an erode operation on the depth-image. We then find the biggest component in the specified depth range. From this component we accumulate the foreground colors. Then we compute the superset by expanding the border pixels of the biggest component and classify everything outside the blob as background. After about 20 frames, we have accumulated enough sample colors. We initialize the SIOX engine with the accumulated colors (compare Figure 8).

6.2 Applying SIOX

We can now apply the SIOX on the superset using colors acquired using the depth range check. By applying it on the superset, we can be sure that SIOX will not classify large portions of the board as foreground if the colors of the teacher image are very similar to other objects in the camera view. New colors are continuously learned during the lecture. Since we have a relatively robust way to find the instructor's superset, we can add "fresh" background colors from everything outside this superset. This is helpful because the board contents and thus its colors are changing continuously over time.

The approach works in real-time with 25 frames per second on a video with 640×480 pixels and is rather resistant against blurred edges and de-synchronization effects. Figure 9 shows some examples of final segmentation results.

Figure 8. Acquiring color sample for the SIOX algorithm. The subset of the teacher is highlighted. We can also easily compute the superset by expanding these pixels (see rectangle).

7 Conclusion

3D time-of-flight cameras in combination with standard 2D-cameras promise an efficient way to solve many segmentation problems. In practice however, the exact calibration and synchronization of the two cameras is tricky. The 3D cameras do not yet provide any explicit synchronization capability, such as those provided by many FireWire cameras. The low resolution and frame rate of the 3D-cameras is not enough for many segmentation tasks to be performed directly. Furthermore, the low signal-to-noise ratio causes many problems. Besides overflows, there are other artifacts caused by quickly moving objects, light scattering, background illumination, or the non-linearity of the measurement. Last but not least, using a time-of-flight camera requires a large budget. This made many researches restrain from a serious use of this technology. This article, however, shows that the cameras already provide an interesting supplement to other segmentation methods. The presented real-live problem could easily be solved by combining the 3D time-of-flight depth information with a software segmentation approach. While 3D cameras improve during the next years they are also becoming cheaper they have the potential to revolutionize segmentation approaches.

Figure 9. Examples of the final segmentation results. These pixel-accurate results can be computed in real-time.

References

[1] 3DV Systems Inc. DMC 100 Depth Machine Camera (last visited: 2005-07-01). http://www.3dvsystems.com, 2004.

[2] D. Beymer, P. McLauchlan, B. Coifman, and J. Malik. A Real-time Computer Vision System for Measuring Traffic Parameters. In *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recog nition (CVPR)*, June 1997.

[3] Y. Boykov and M.-P. Jolly. Interactive Graph Cuts for Optimal Boundary and Region Segmentation of Objects in N-D Im ages. In *Proceedings of the International Conference on Computer Vision*, pages 105–112, Vancouver, Canada, July 2001.

[4] Canesta Inc. CanestaVision EP Development Kit (last visited: 2005-07-01). http://www.canesta.com/devkit.htm, 2004.

[5] S.-Y. Chien, Y.-W. Huang, S.-Y. Ma, and L.-G. Chen. Automatic Video Segmentation for MPEG-4 using Predictive Watersheds. In *Proceedings of IEEE International Conference on Multimedia and Expo*, pages 239–243, Tokyo, Japan, August 2001.

[6] CSEM Sa. SwissRanger 3D Vision Camera (last visited: 2005-07-01). http://www.swissranger.ch, 2004.

[7] J. Diebel and S. Thrun. An application of markov random fields to range sensing. In *Proceedings of Conference on Neural Information Processing Systems (NIPS)*, Cambridge, MA, USA, 2005. MIT Press.

[8] G. Friedland, K. Jantz, L. Knipping, and R. Rojas. Image Segmentation by Uniform Color Clustering – Approach and Benchmark Results. Technical Report B-05-07, Freie Universität Berlin, Institut ür Informatik, June 2005.

[9] G. Friedland, K. Jantz, and R. Rojas. Siox: Simple interactive object extraction in still images. In *Proceedings of the IEEE Symposium on Multimedia (ISM2005)*, pages 253–259, Irvine, California, December 2005.

[10] G. Friedland, T. Lenz, K. Jantz, and R. Rojas. Extending the SIOX Algorithm: Alternative Clustering Methods, Sub-pixel Accurate Object Extraction from Still Images, and Generic Video Segmentation. Technical Report B-06-06, Freie Universität Berlin, Institut ür Informatik, January 2006.

[11] S. Gibbs, C. Arapis, C. Breiteneder, V. Lalioti, S. Mostafawy, and J. Speier. Virtual Studios: An Overview. *IEEE Multimedia*, 5(1):18–35, January–March 1998.

[12] S. B. Göktürk, hakan Yalcin, and C. Bamji. A Time-Of-Flight Depth Sensor - System Description, Issues and Solutions. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, Washington D.C., USA, July 2004.

[13] S. B. Göktürk and C. Tomasi. 3D Head Tracking Based on recognition and Interpolation Using a Time-Of-Flight Depth Sensor. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*, Washington D.C., USA, July 2004.

[14] G. Gordon, T. Darrel, M. Harville, and J. Woodfill. Background estimation and removal based on range and color. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, Fort Collins, CO, USA, June 1999.

[15] I. Haritaoglu, D. Harwood, and L. Davis. W4: Real-Time Surveillance of People and Their Activities. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(8):809–831, August 2000.

[16] L. Li, W. Huang, I. Y. H. Gu, and Q. Tian. Foreground Object Detection from Videos Containing Complex Background. In *Proceedings of ACM Multimedia 2003*, Berkeley, California, USA, November 2003.

[17] X. Luan, R. Schwarte, Z. Zhang, Z. Xu, H.-G. Heinol, B. Buxbaum, T. Ringbeck, and H. Hess. Three-dimensional intelligent sensing based on the PMD technology. *Sensors, Systems, and Next-Generation Satellites V. Proceedings of the SPIE.*, 4540:482–487, December 2001.

[18] R. Mertens, G. Friedland, and M. Krüger. To See or Not To See: Layout Constraints, the Split Attention Problem and their Implications for the Design of Web Lecture Interfaces. In *Proceedings of the AACE E-Learn - World Conference on E-Learning in Corporate, Government, Healthcare, and Higher Education*, Honolulu (Hawai), USA, October 2006.

[19] T. Oggier, M. Lehmann, R. Kaufmann, M. Schweizer, M. Richter, P. Metzler, G. Lang, F. Lustenberger, and N. Blanc. An all-solid-state optical range camera for 3D real-time imaging with sub-centimeter depth resolution (SwissRanger). *Optical Design and Engineering. Proceedings of the SPIE.*, 5249:534–545, Februar 2004.

[20] PMD Technologies GmbH. PMDTec 3D Vision Camera (last visited: 2005-07-01). http://www.pmdtec.com, 2004.

[21] R. Rojas, G. Friedland, L. Knipping, and E. Tapia. Teaching with an intelligent electronic chalkboard. In *Proceedings of ACM Multimedia 2004, Workshop on Effective Telepresence*, pages 16–23, New York, New York, USA, October 2004.

[22] C. Rother, V. Kolmogorov, and A. Blake. "grabcut": interactive foreground extraction using iterated graph cuts. *ACM Trans. Graph.*, 23(3):309–314, 2004.

[23] M. Simon, S. Behnke, and R. Rojas. Robust real time color tracking. In *RoboCup 2000: Robot Soccer World Cup IV*, pages 239–248, Heidelberg, Germany, 2001. Springer.

[24] J. Weingarten, G. Gruener, and R. Siegwart. A State-of-the-Art 3D Sensor for Robot Navigation. In *Proceedings of IROS 2004*, 2004.

[25] C. L. Zitnick and T. Kanade. A Cooperative Algorithm for Stereo Matching and Occlusion Detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(7):675–684, July 2000.