

# Fast Lighting Independent Background Subtraction

Yuri Ivanov<sup>1</sup>, Aaron Bobick<sup>2</sup> and John Liu<sup>1</sup>

<sup>1</sup>*MIT Media Laboratory, 20 Ames St, E15-368A, Cambridge, MA 02139*

<sup>2</sup>*Georgia Institute of Technology, College of Computing, 801 Atlantic Dr., Atlanta, GA 30332*

**Abstract.** This paper describes a simple method of fast background subtraction based upon *disparity verification* that is invariant to arbitrarily rapid run-time changes in illumination. Using two or more cameras, the method requires the off-line construction of disparity fields mapping the primary background images. At runtime, segmentation is performed by checking background image to each of the additional auxiliary color intensity values at corresponding pixels. If more than two cameras are available, more robust segmentation can be achieved and, in particular, the *occlusion shadows* can be generally eliminated as well. Because the method only assumes fixed background geometry, the technique allows for illumination variation at runtime. Since no disparity search is performed, the algorithm is easily implemented in real-time on conventional hardware.

**Keywords:** background subtraction, image segmentation, stereo, disparity warp

## 1. Introduction: Background Subtraction

Perhaps the most frequently solved problem in computer vision is segmenting a foreground object from its background in an image. Recent work in video surveillance, intelligent environments and perceptual user interfaces (Brill et al., 1998; Darrell et al., 1994; Intille et al., 1997) involves vision systems which interpret the pose or gesture of users in a known, indoor environment. In particular, video surveillance applications often have to perform in an environment where the background is geometrically static. As we show later in this paper, the constraint of the geometrically static background makes it easy to accommodate rapid lighting changes for the purposes of the background subtraction.

There also exist applications which could greatly benefit from using computer vision algorithms, but have traditionally been found too difficult for the computer vision to handle for the reason of high variability of lighting conditions. One such application is a computer theater (Pinhanez and Bobick, 1999). In theatrical performances the lighting serves expressive purposes and should not be as restrictively controlled as in other applications where lighting change is considered a nuisance. Such applications can take advantage of the known background geometry in order to provide the higher degree of lighting independence. They can be generally characterized by the following:



© 2000 Kluwer Academic Publishers. Printed in the Netherlands.

1. there are frequent unpredictable changes in lighting;
2. the background geometry is static;
3. multiple cameras are available.

The goal of this paper is to develop a background subtraction method that performs well in such situations.

Most common approach to segmenting objects from the background is some form of background subtraction. For example, in (Wren et al., 1997; Stauffer and Grimson, 1999; Friedman and Russell, 1997) authors use statistical texture properties of the background observed over extended period of time to construct a model of the background, and use this model to decide which pixels in an input image do not fall into the background class. The fundamental assumption of the algorithm is that the background is static, or at most slowly varying, in all respects: geometry, reflectance, and illumination. These algorithms show low sensitivity to slow lighting changes, to which they need to adapt.

(Oliver et al., 1999) uses a subtraction method which has an explicit illumination model. The model in this case is the eigenspace, describing a range of appearances of the scene under a variety of lighting conditions. The method works extremely well for outdoor environments with static geometry, but unforeseen illumination conditions, which are not taken into account while building the model, will degrade the performance of the algorithm.

All above algorithms are not designed for handling rapid lighting changes, such as theatrical lights or a still camera flash, with which our technique is concerned.

In contrast, the approach most related to the technique presented in this paper is based upon geometry. (Gaspar et al., 1994) use geometrical constraints of a ground plane in order to detect obstacles on the path of a mobile robot. (Okutomi and Kanade, 1993) employ special purpose multi-baseline stereo hardware to compute dense depth maps in real-time. Provided with a background disparity value, the algorithm can perform real-time depth segmentation or “z-keying” (Kanade, 1995). The only assumption of the algorithm is that the geometry of the background does not vary. However, the computational burden of computing dense, robust, real-time stereo maps requires great computational power.

In this paper we present a fast, simple method for segmenting people from a geometrically static background. In the remainder of this paper we describe the algorithm in detail, present experimental results, and discuss implementation details critical to the performance of the technique.

## 2. Our Approach

The background subtraction method presented in this paper classifies image points by their belonging to a known static surface, rather than to a group of neighboring points of similar texture. Unfortunately, the direct implementation of this idea requires executing dense stereo algorithms in real time, which is possible, but requires either massive computational power or specialized hardware, as in (Okutomi and Kanade, 1993; Kanade, 1995).

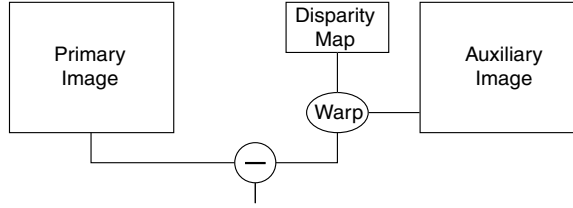
The main insight of this paper is that, for the purposes of background subtraction, we can avoid the on-line computation of depth and the reconstruction of the 3D model of space at each step. If the background is geometrically static, stereo disparity between a *primary* and an *auxiliary* camera views of the background scene is also static fully specifying the pixel-to-pixel transformation from one image of empty scene to another. Ideally, this model is only violated by an object that does not belong to any of the background surfaces. After the disparity map for an empty scene is built off-line we look for pixels violating this model in each new incoming frame .

The basic background disparity verification algorithm can be described as follows - for each pixel in the *primary* image:

1. Using the disparity warp map find the pixel in the *auxiliary* image, which corresponds to the current pixel;
2. If the two pixels have the same color and luminosity, label the primary image pixel as background;
3. If the pixels have different color or luminosity, then the pixel in the primary image either belongs to a foreground object, or to an “occlusion shadow”: a region of the primary image which is not seen in the auxiliary camera view due to the presence of the actual object;
4. If multiple cameras are available, verify the potential object pixels by warping to each of the other auxiliary images and looking for background matches.

The algorithm is illustrated in Figure 1. Because the basis of comparison is the background disparity warp between two images taken at the same time, illumination or, to a certain degree, reflectance can vary without significantly affecting the performance.

It should be noted that all that is required for the algorithm to work is a disparity map between the primary image and each of the



*Figure 1.* Illustration of the proposed subtraction algorithm. The algorithm first warps the auxiliary image according to the disparity map and then subtracts the result from the primary camera view. The remaining pixels either belong to a foreground object or are occluded from the auxiliary camera.

auxiliary images. Of course, as in any matching based technique, photometric variation between cameras should be minimized for optimal performance.

An important assumption that we use is that the surface reflectance properties are close to Lambertian. The algorithm can be adjusted for any other kind of reflectance, but the general property has to be known in advance in order to relate intensity between pixels in each camera view. Later in the exposition we give one solution to a special case where part of the background is clearly non-Lambertian.

The complete method consists of the three stages: computing the off-line disparity map(s), verifying matching background intensities, and eliminating occlusion shadows. We detail each presently.

## 2.1. NOTATION

For the rest of the paper we will assume the following notations: let  $\mathbf{r}$  be a coordinate vector  $(x, y)$  in the primary image  $\mathbf{I}$ , and let  $\mathbf{r}'$  be a vector of transformed coordinates  $(x', y')$  in the auxiliary image  $\mathbf{I}'$ . Then:

$\mathbf{r} = (x, y)$	position in primary image
$\mathbf{I}(\mathbf{r})$	primary image pixel
$D^x(\mathbf{r})$	horizontal disparity
$D^y(\mathbf{r})$	vertical disparity
$x' = x - D^x(\mathbf{r})$	x position in auxiliary image
$y' = y - D^y(\mathbf{r})$	y position in auxiliary image
$\mathbf{r}' = (x', y')$	position in auxiliary image
$\mathbf{I}'(\mathbf{r}')$	auxiliary image pixel

where vector  $\mathbf{I}(\mathbf{r})$  consists of color components of the pixel (R, G, and B in RGB color space, Y, U and V in YUV color space, etc.).

When convenient, we will consider the  $x$ - and  $y$ - disparity maps,  $D^x(\mathbf{r})$  and  $D^y(\mathbf{r})$ , jointly, denoting it by  $\mathbf{D}(\mathbf{r})$ .

## 2.2. BUILDING THE MODEL

In order for the algorithm to work, a dense disparity map needs to be built to define the transformation between the primary and auxiliary camera views. Since the disparity model for the scene is built off-line, any available technique can be used to compute it automatically. However, in indoor environments with large texture-less regions (often black matte in the case of a theater) most stereo algorithms will experience difficulties estimating the disparity accurately. The technique which we found to be the most convenient is to first incrementally build a robust sparse disparity map by an active point-scanning and then interpolate and refine it as described below.

We further motivate this approach by the necessity to have a disparity map which is reasonably smooth but allows for sharp discontinuities (e.g. hanging projection screens, as shown in the results section). This requires some sort of a piecewise smooth interpolation scheme. We found it convenient to define the surface by a set of sample points at which the disparity is easily computed. Delaunay triangulation then lets us connect these samples into a set of patches and interpolate each patch by a smooth polynomial under the assumption that the surface inside each patch is smooth. This approach allows for sharp discontinuities in disparity as well as for a variable rate of precision throughout the scene. An additional advantage of this technique is that the effect of outliers is localized and only affects the immediate neighbors of the erroneous point.

The steps necessary for building the background disparity model are detailed below:

### 1. Data collection.

Using a laser pointer to cast an easily detectable point on the background surface, we record the horizontal and vertical disparities at arbitrarily selected sparse points. We use simple visualization tools that let us easily detect outliers and correct for them by just “painting over” the suspicious areas - that is, scanning the laser over these areas to record more samples. These tools also help increase the density of the points around the edges of the background objects to improve precision of the disparity map in those areas.

### 2. Interpolation.

We construct a dense disparity map by piecewise interpolation. The irregular grid of measured points is converted into a set of neighboring triangular patches by performing Delaunay triangulation of the set, and then interpolate each patch of the mesh using the established neighborhood relationships. Despite the fact that the

disparity is inversely proportional to depth, if the size of each patch is small relative to the depth and the length of the baseline, then the disparity can be safely interpolated by a planar patch<sup>1</sup>. The advantage of this patch-wise interpolation is that the effect of outliers is localized to the area surrounded by their immediate neighbors. Note that this local interpolation scheme does not assume any particular background geometry (e.g. large planar surfaces).

### 3. Refinement (optional)

In regions of the image where there is sufficient texture we can optionally compensate for possible errors that have been introduced at each of the previous steps (e.g. imprecise location of the centroid of the laser spot, or the integer quantization of our point correspondence algorithm). An iterative procedure refines the interpolated disparity map by first using that map to get a “rough” approximation of the corresponding location of a pixel from the primary image in the auxiliary image. Then a search over a small window centered around the current location in the auxiliary view is performed to find the best local match. This location of the new best match defines the new value for the disparity map at pixel  $\mathbf{r}$ . The process is iterated until no further refinement is made. Starting with  $\mathbf{D}_0(\mathbf{r}) = \mathbf{D}(\mathbf{r})$ , we compute  $i$ -th refinement of the map as follows:

$$\mathbf{D}_i(\mathbf{r}) = \mathbf{r} - \arg \min_{\mathbf{q}_i \in \mathcal{W}} \|\mathbf{I}(\mathbf{r}) - \mathbf{I}(\mathbf{q}_i)\|^2 \quad (1)$$

where  $\mathcal{W}$  is the set of pixel locations around  $\mathbf{r}_i$  within a window of size  $\mathbf{W}$ :

$$\mathcal{W} = \{\mathbf{q}_i : \mathbf{q}_i \in [\mathbf{r}'_i - \mathbf{W}/2; \mathbf{r}'_i + \mathbf{W}/2]\}$$

and  $\mathbf{r}'_i$  is computed from the previous estimation of disparity:

$$\mathbf{r}'_i = \mathbf{r} - \mathbf{D}_{i-1}(\mathbf{r})$$

for  $i = 1, 2, \dots$

---

<sup>1</sup> In our experiments the depth was  $> 10$  meters with the baseline of about 2 meters and the patch size on the order of centimeters

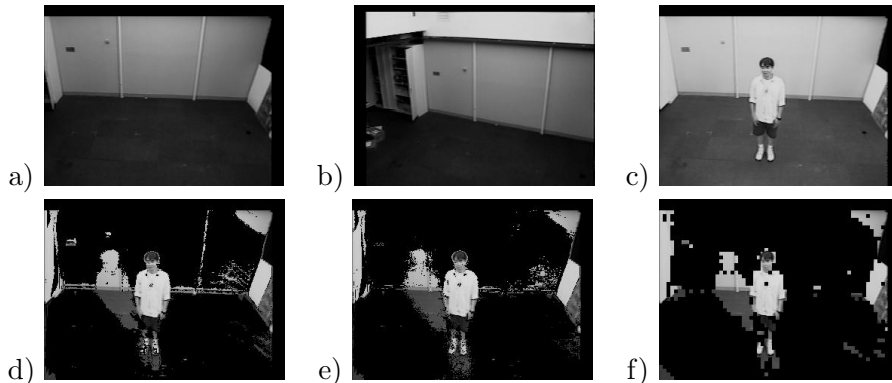


Figure 2. Background subtraction algorithm. a) primary camera view of the empty scene, b) auxiliary camera view of an empty scene, c) primary camera view with a person in the scene. Bottom row: (d) results using tolerance range, (e) results using tolerance range on refined disparity map, (f) windowed means on refined disparity map.

### 2.3. SUBTRACTION

The subtraction algorithm proceeds by warping each pixel of the primary image to the auxiliary image positions and comparing the color and luminosity values<sup>2</sup>. We need to define a binary masking function  $f(\mathbf{r})$  for construction of the background subtracted image. The masking function  $f(\mathbf{r})$  determines the complexity and accuracy of the algorithm.

In general,  $f(\mathbf{r})$  is a boolean function which takes a value of 1 for all primary image locations,  $\mathbf{r}$ , which belong to the set of foreground pixels,  $\mathcal{F}$ :

$$f(\mathbf{r}) = \begin{cases} 1, & \forall \mathbf{r} \in \mathcal{F} \\ 0, & \forall \mathbf{r} \in \bar{\mathcal{F}} \end{cases} \quad (2)$$

Then, the subtracted image is formed simply by applying the mask to the primary camera view:

$$\mathbf{S}(\mathbf{r}) = f(\mathbf{r})I(\mathbf{r}) \quad (3)$$

where  $\mathbf{S}(\mathbf{r})$  is the resulting image with the background pixels removed.

In its simplest form the foreground pixel set,  $\mathcal{F}$ , is just a set of pixels not satisfying the disparity constraint. In this case the function  $f(\mathbf{r})$  is determined as follows:

<sup>2</sup> It is a minor discrepancy with the approach described so far (figure 1), where the direction of the warp is chosen to be from auxiliary image to the primary one. The direction in which the warping is performed is inconsequential to the algorithm and in the diagram is chosen only to simplify the presentation.

$$f(\mathbf{r}) = \begin{cases} 0 & \text{if } \mathbf{I}(\mathbf{r}) = \mathbf{I}'(\mathbf{r}') \\ 1 & \text{otherwise} \end{cases} \quad (4)$$

In reality we never get measurements good enough to comply with this sort of screening, so we relax this constraint to compensate for possible errors and formulate the comparison to accept a value within some tolerance range  $\epsilon$ :

$$f(\mathbf{r}) = \begin{cases} 0 & \text{if } |\mathbf{I}(\mathbf{r}) - \mathbf{I}'(\mathbf{r}')| < \epsilon \\ 1 & \text{otherwise} \end{cases} \quad (5)$$

Along with this straightforward method of subtraction, we have implemented a more robust sub-sampling method which performs subtraction over a small neighborhood of a pixel. This technique introduces the least amount of interference with the normal computations, while giving the advantage of being more forgiving about the precision of the disparity map. We partition the original primary image into a grid of small windows and compute a mean color intensity of each window of the grid. Next we warp the pixels of each window of the primary image into the auxiliary image and compute the mean color intensity of the resulting set of generally non-contiguous points. We then compare the means by the same procedure as above. For a  $k \times k$  window of pixels in the primary image  $W_{ij}$ , the window mean images are computed as:

$$\bar{\mathbf{I}}_{ij} = \frac{1}{k^2} \sum_{x,y \in W_{ij}} \mathbf{I}(\mathbf{r}) \quad (6)$$

$$\bar{\mathbf{I}}'_{ij} = \frac{1}{N} \sum_{x',y' \in W'_{ij}} \mathbf{I}'(\mathbf{r}') \quad (7)$$

where  $W'_{ij}$  is the set of auxiliary image pixel locations to which the pixels in the primary image window were warped.  $N$  in the second equation denotes the number of pixels in the auxiliary image that after warping remained inside the auxiliary image boundaries. The masking function is defined as:

$$f(\mathbf{r}) = \begin{cases} 0 & \text{if } |\bar{\mathbf{I}}_{ij} - \bar{\mathbf{I}}'_{ij}| < \epsilon, \\ & x, y \in W_{ij}; x', y' \in W'_{ij} \\ 1 & \text{otherwise} \end{cases} \quad (8)$$

which yields a blocked (sub-sampled) background subtracted image.

The proposed algorithm is simple and fast enough for real time performance. It is as fast as the conventional image differencing routines used, for example, in (Intille et al., 1997) in that each pixel is only

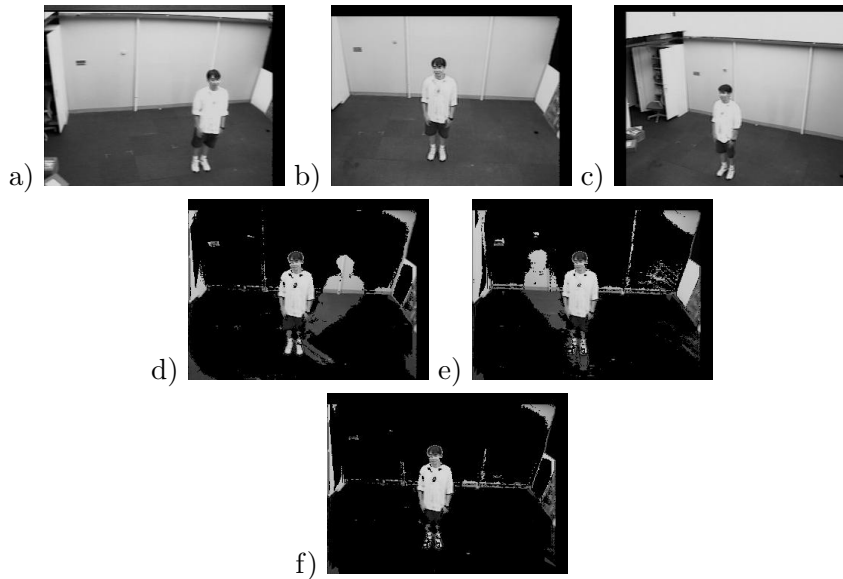


Figure 3. Removing the occlusion shadow using an additional auxiliary camera. a) left auxiliary camera view, b) primary camera view, c) right auxiliary camera view, d) subtraction using views from a) and b), e) subtraction using views from b) and c), f) results of the removal of the occlusion shadow.

examined once. One obvious advantage is that in using disparity for identifying a candidate background, we can accommodate for changing textures and lighting conditions<sup>3</sup>. The algorithm effectively removes shadows and can be modified to remove surfaces selectively.

#### 2.4. REMOVING OCCLUSION SHADOWS

The occlusion shadow is the region of pixels in the primary camera view, which is not seen in the auxiliary view due to occlusion. The effect of occlusion shadow is clearly seen in figures 2 and 3. Intuitively, the occlusion shadow is of the same shape as a regular shadow that would be cast by the object if the auxiliary camera were a light-source.

Having additional cameras allows us to remove occlusion shadows by cross-verifying each pixel across several camera views. As shown in equation 2, for a location  $\mathbf{r}$  in the primary camera view each  $l$ -th auxiliary camera provides a boolean function,  $f_l(\mathbf{r})$ , which is set to 1 if the corresponding pixel belongs to a set of foreground pixels,  $\mathcal{F}_l$ , and 0 otherwise:

<sup>3</sup> We presume that cameras are synchronized and take images simultaneously. For this purpose genlocked cameras are typically used with the auto-gain mechanism disabled.

$$f_l(\mathbf{r}) = \begin{cases} 1, & \forall \mathbf{r} \in \mathcal{F}_l \\ 0, & \forall \mathbf{r} \notin \mathcal{F}_l \end{cases} \quad (9)$$

The true “object” pixel will violate the background disparity in all camera pairs, whereas the occlusion shadow (the area occluded by an object in the auxiliary camera view) gets marked as background in other views. This observation lets us form a function which assigns a pixel to fore- or a background across multiple views:

$$f(\mathbf{r}) = \bigwedge_l f_l(\mathbf{r}) \quad (10)$$

The function  $f(\mathbf{r})$  takes a value of 1 when the pixel at  $\mathbf{r}$  belongs to the foreground in every camera pair:

$$f(\mathbf{r}) = \begin{cases} 1, & \forall \mathbf{r} \in \mathcal{F} = \bigcap_l \mathcal{F}_l \\ 0, & \forall \mathbf{r} \notin \mathcal{F} \end{cases} \quad (11)$$

The function  $f(\mathbf{r})$  is applied to the primary image<sup>4</sup> as shown in equation 3.

The choice of the length of the baseline has an effect on the performance of the algorithm. With a short baseline, the occlusion shadows are minimal, but the algorithm does not perform very well, since the disparity map will be very sensitive to noise and computational errors.

On the other hand, the long baseline makes the subtraction algorithm more robust, but the occlusion shadows increase and have to be removed with additional cameras.

### 3. Practical considerations

Implementation of the proposed algorithm is straightforward. However, as in all stereo problems, special care should be given to minimizing the errors resulting from photometric variation of the main and auxiliary cameras and maintaining the accuracy of the disparity map. This is particularly true for some of our target applications where surfaces may not be purely Lambertian resulting in different image intensities when viewed from different directions.

---

<sup>4</sup> This choice of  $f(\mathbf{r})$  has a potential danger of eroding the object in case of erroneous disagreement between different camera pairs if more than two are used. This suggests using some sort of voting threshold to make the final decision.

### 3.1. REDUCING PHOTOMETRIC VARIATIONS

Photometric variation is a concern for pixel-to pixel comparisons as is performed in this algorithm. It arises from two sources:

1. difference in gain and offset between cameras (un-calibrated photometry);
2. background surfaces with non-Lambertian reflectance properties.

When the cameras are not color-calibrated simple subtraction of the pixel values would not work. Correction for the photometric variations between the cameras requires estimating the  $YUV$  mapping from one camera to the other.

We estimate parameters of the transformation from a set of examples of uniformly colored patches for each camera pair and represent the mapping by a best-fit polynomial. Our experience showed that a simple linear model fits the training data well for most cases, and often a single map can be used for the entire image.

For our application the only significant exception to the Lambertian assumption were the rear-projection video screens. Unlike Lambertian surfaces where light emission falls off by the cosine of the angle between the emission direction and the surface normal, the light emitted by these screens falls off much more sharply. The result is that the surface appears darker when observed from more oblique angles. To compensate for such effects, we allow the linear photometric correction parameters to be indexed by image location, defining different maps for different image regions. We assume that the effect is insensitive to color, and use the same map for  $Y$ ,  $U$ , and  $V$ .

Specularities present a problem for our algorithm. Due to instability of specularities purely intensity-based or stereo-based methods will experience the same difficulty.

## 4. Experimental results

Results of our algorithm are shown in Figures 2, 3 and 4. In the Figure 2 the first pair of images, Figures 2a and 2b show two camera views of an empty scene; Figure 2c is the primary image at run time with the person in the view of the camera. The background is subtracted by one of the following methods (Figure 2d-f): 1) using the tolerance range  $\epsilon$  (eqn. 5), 2) using the tolerance range  $\epsilon$  on a disparity map adjusted by the refinement procedure (eqns. 1 and 5), and 3) using windowed

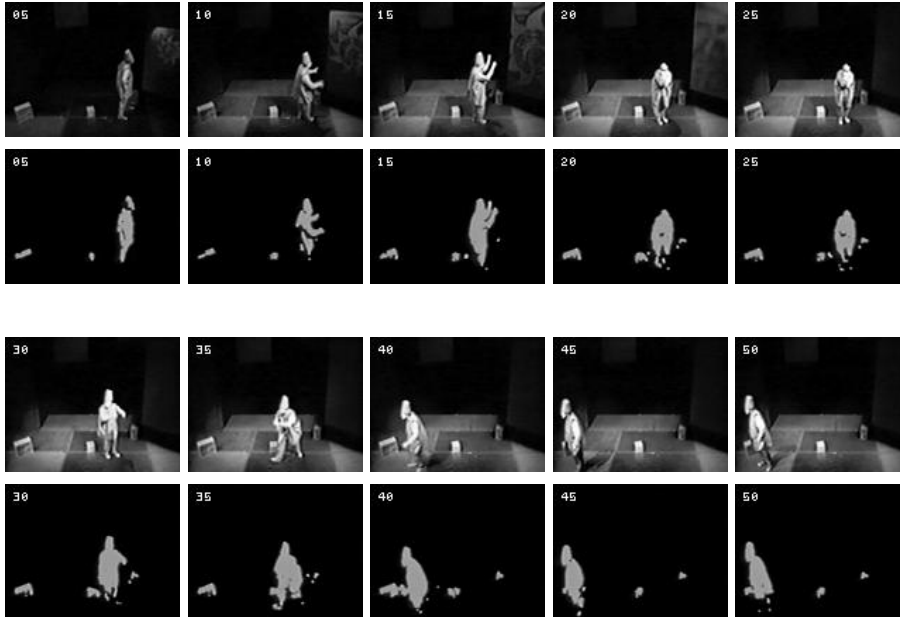


Figure 4. Results of the algorithm during the computerized theatrical performance. The images show a sequence of ten video frames (1st and 3rd rows of images) , taken from a video five frames apart, and results of background subtraction by the proposed technique (2nd and 4th rows). Changes in lighting and background texture can be seen.

means on a refined disparity map (eqns. 1 and 8). Note the presence of the occlusion shadow on the left behind the person.

Figure 3 demonstrates the process of removing the occlusion shadow. Three camera views are shown in the top row, where Figure 3a is a left auxiliary camera view, Figure 3b is a primary camera view and Figure 3c is a right auxiliary camera view. Figures 3d and 3e show subtraction performed on two pairs of images. Figure 3f shows the result of using all three cameras.

The proposed technique has been used in computerized theatrical performance “It/I” (Pinhanez and Bobick, 1999). A computer vision system was used to control a virtual character which interacted with a human actor. Theatrical lighting and projection screens presented a problem for traditional segmentation techniques which had to be revised for our application.

During the performance we found that sometimes we encounter situations where high intensity of the lights leads to saturation of regions in the image. In those regions, pixels are “washed out” and provide no useful color information; the  $U$  and  $V$  values are unreliable. In other cases, the lighting is so low that it falls below the camera’s

sensitivity and provide no information at all. For these situations we employ simple modifications to the original masking function presented in equation (5), which are easy to implement in YUV color space: if the value of  $Y$  component is above some threshold value  $Y_{max}$ , we base the comparison  $|\mathbf{I}(\mathbf{r}) - \mathbf{I}'(\mathbf{r}')| < \epsilon$  only on  $Y$  component of the vector. If the value of the  $Y$  component falls below some sensitivity threshold  $Y_{min}$ , the pixel is labeled as a background. And, finally, for all other values of  $Y$ , equation (5) is used as given.

The results of application of the proposed algorithm are shown in figure 4, where the disparity-based subtraction technique described in this article effectively compensates for dramatic changes in lighting and background texture. In the sequence presented in the figure, the lighting changes in both intensity and color. The starting frame shows the scene under a dim blue light. In a time interval of about two seconds the lighting changes to bright warm Tungsten yellow. In frames 5 through 20 the projection screen shows an animation which is effectively removed by our algorithm. The algorithm is very fast and during the performance delivered a frame rate of approximately 14  $320 \times 240$  pixel frames per second running on a 180 MHz SGI Indy R5000.

## 5. Conclusions/Future work

In this paper we described a new approach to background subtraction, which is lighting insensitive and suitable for real-time applications. We proposed the use of simplified stereo algorithm to perform the segmentation, which requires at least two camera views to be available. We demonstrated that the occlusion problem can be dealt with easily if more than two cameras are available.

Further work might be required to refine the shown method of segmentation. The accuracy of the algorithm is related to the accuracy of the disparity map we obtain during the first, off-line, part of the algorithm. As was discussed above, any available method of building the disparity map can be used. The method which is briefly described in this paper delivered sufficient accuracy for our application and was successfully used in a theatrical performance.

## References

- Brill, F. Z., T. J. Olson, and C. Tserng: 1998, 'Event Recognition and Reliability Improvements for the Autonomous Video Surveillance System'. In: *Image Understanding Workshop*. Monterey, CA, pp. 267–283.

- Darrell, T., P. Maes, B. Blumberg, and A. Pentland: 1994, 'A Novel Environment for Situated Vision and Behavior'. In: *Proc. of CVPR-94 Workshop for Visual Behaviors*. Seattle, Washington, pp. 68–72.
- Friedman, N. and S. Russell: 1997, 'Image segmentation in video sequences: A probabilistic approach'. In: *Thirteenth Conference on Uncertainty in Artificial Intelligence*.
- Gaspar, J., J. Santos-Victor, and J. Senteiro: 1994, 'Ground Plane Obstacle Detection with a Stereo Vision System'. *International workshop on Intelligent Robotic Systems*.
- Intille, S. S., J. W. Davis, and A. F. Bobick: 1997, 'Real-Time Closed-World Tracking'. In: *Proc. of CVPR-97*. San Juan, Puerto Rico, pp. 697–703.
- Kanade, T.: 1995, 'A Stereo Machine for Video-Rate Dense Depth Mapping and Its New Applications'. In: *Proc. of Image Understanding Workshop*. Palm Springs, California, pp. 805 – 811.
- Okutomi, M. and T. Kanade: 1993, 'A Multiple-Baseline Stereo'. *IEEE Trans. on Pattern Analysis and Machine Intelligence* **15**(4), 353–363.
- Oliver, N., B. Rosario, and A. Pentland: 1999, 'A Bayesian Computer Vision System for Modeling Human Interactions'. In: *Proceedings of ICVS99*. Gran Canaria, Spain.
- Pinhanez, C. S. and A. F. Bobick: 1999, 'It/I': A Theater Play Featuring an Autonomous Computer Graphics Character'. In: *Proc. of the ACM Multimedia'98 Workshop on Technologies for Interactive Movies*. Bristol, England, pp. 22–29.
- Stauffer, C. and W. Grimson: 1999, 'Adaptive background mixture models for real-time tracking'. In: *Proc. of CVPR-99*. Ft. Collins, CO, pp. 246–252.
- Wren, C., A. Azarbayejani, T. Darrell, and A. Pentland: 1997, 'Pfinder: Real-Time Tracking of the Human Body'. *IEEE Trans. on Pattern Analysis and Machine Intelligence* **19**(7), 780–785.