M.I.T Media Laboratory Perceptual Computing Section Technical Report No. 437
Appears in:

Fast Lighting Independent Background Subtraction

Yuri Ivanov Aaron Bobick John Liu
[yivanov bobick johnliu]@media.mit.edu
The Media Laboratory
Massachusetts Institute of Technology
20 Ames St., Cambridge, MA 02139

Abstract

This paper describes a new method of fast background subtraction based upon disparity verification that is invariant to run-time changes in illumination. Using two or more cameras, the method requires the off-line construction of disparity fields mapping the primary (or *key*) background image to each of the additional *reference* background images. At runtime, segmentation is performed by checking 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. And, because no disparity search is performed at run time, the algorithm is easily implemented in real-time on conventional hardware.

1 Introduction: Background subtraction

As computer vision begins to address the visual interpretation of *action* [1] applications such as surveillance and monitoring are becoming more relevant. Similarly, recent work in intelligent environments and perceptual user interfaces [2, 4] involve vision systems which interpret the pose or gesture of users in a known, indoor environment. In all of these situations the first fundamental problem encountered is the extraction of the image region corresponding to the person or persons in the room.

Previous attempts at segmenting people from a known background have taken one of three approaches. Most common is a some form of background subtraction. For example, [8] uses 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 in all respects: geometry, reflectance, and illumination.

The second class of approach is based upon image motion only presuming that the background is stationary or at most slowly varying, but that the person is moving. [3]. In these methods no detailed model of the background is required. Of course, these methods are only appropriate for the direct interpretation of motion; if person stops moving, no signal remains to be processed. This method also requires constant or slowly varying geometry, reflectance, and illumination.

The final approach, and the one most related to the technique presented in this paper, is based upon geometry. Kanade, *et al.* [7] 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" [6]. 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. Using two or more cameras and background disparity maps created off-line, segmentation is performed by checking color intensity values at corresponding pixels: if the values match, the background disparity is validated and the pixel in the key image is assumed to belong to the background. Otherwise, it is labeled as object. Because the basis of comparison is background disparity warp between two images taken at the same time, illumination or reflectance can vary without significantly affecting the results.

In the remainder of this paper we describe the algorithm in detail, present some experimental results, and discuss implementation details critical to the performance of the technique.

2 Our approach

As noted in [5], there have been very few efforts to use three-dimensional models of objects and knowledge of image formation in refining segmentation. The background subtraction method presented in this paper addresses this observation, classifying 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 either requires massive computational power or, as in work of Kanade, requires special hardware [7, 6].

The main insight of this paper is that, for the purposes of background subtraction, we can avoid the on-line computation of depth altogether and the reconstruction of the 3D model of space at each step. The idea is simple: build a model of the empty space off-line and then, looking at each incoming frame, scan through the image for pixels that do not comply with this model.

The basic background disparity verification algorithm can be described as follows: For each pixel in the *key* image:

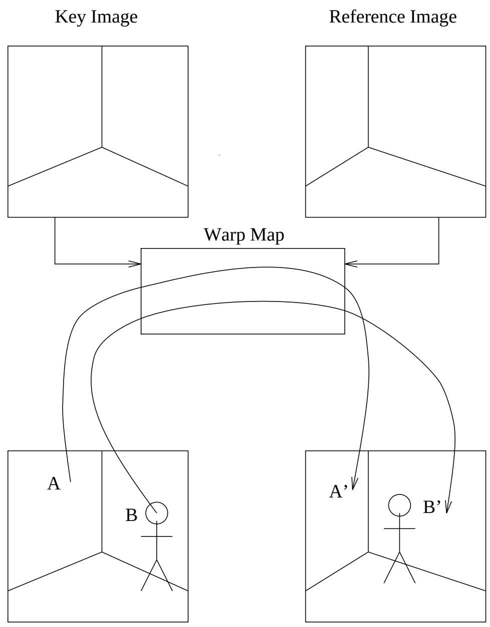

Figure 1: Illustration of the subtraction algorithm. Warp map is created off-line by processing a stereo pair of the empty scene. During the subtraction phase, point A is warped to A' and is removed. Point B is warped to B' and is not removed because it does not satisfy the disparity map.

1. Warp the pixel of the key image into a corresponding pixel of the *reference* image.
2. If the reference pixel has the same color and luminosity as the key pixel, label the key pixel as background.
3. If the reference pixel has different color and luminosity than the key pixel, then the key pixel is either a foreground object, or in an "occlusion shadow": a region of background key pixels that can not be seen by the reference camera because of the presence of the actual object.
4. If multiple cameras are available, verify the potential object pixels by warping to each of the other reference images and looking for background matches.

The algorithm is illustrated in Figure 1.

It should be noted that no geometrical camera calibration is required for the algorithm to work. All that is required is a disparity map between the key image and each of the reference images. Of course, as in any matching based technique, photometric variation between cameras should be minimized for optimal performance.

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

2.1 Building the model

As an off-line part of the algorithm, we need to build a disparity model of the background. We can, of course, perform traditional full stereo processing. However, because we are typically in an indoor environment with large texture-less regions, and because we have no computational constraints, we use an active point-scanning approach to first get a robust sparse map, and then we interpolate and refine that map. In detail the steps are:

1. **Data collection.**
   Using a laser pointer to cast an easily detected point on the background surface, we record the horizontal and vertical disparities at arbitrarily selected sparse points.
2. **Interpolation.**
   Construct a dense disparity map by piecewise interpolation. We convert the irregular grid of measured points 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. (See appendix A.) In this work, we found that linear patch interpolation gives results that are within the tolerance range of the whole method.
3. **Refinement** (optional)
   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 simple 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 pixel from the key in the reference image. Then a small search is performed to find the best local match. This location of the new best match becomes the new disparity.

2.2 Subtraction

As was described above, the subtraction algorithm proceeds by warping the pixels in the key image into the reference images and comparing the color intensity values. Let:

(x, y)	position in key image
$\mathbf{K}(x, y)$	key image pixel
$D_x(x, y)$	horizontal disparity
$D_y(x, y)$	vertical disparity
$x' = x - D_x(x, y)$	corresponding position in reference image
$y' = y - D_y(x, y)$	corresponding position in reference image
$\mathbf{R}(x', y')$	reference image pixel

We need to define a binary masking function $m(\mathbf{K}(x,y), \mathbf{R}(x',y'))$ for construction of the background subtracted image. The masking function $m(\cdot,\cdot)$ determines the complexity and accuracy of the algorithm.

In its simplest form function $m(\mathbf{K}(x,y), \mathbf{R}(x',y'))$ is just a set of pixels not satisfying the disparity constraint:

$$m(\mathbf{K}(x,y), \mathbf{R}(x',y')) = \begin{cases} 0 & \text{if } \mathbf{K}(x,y) = \mathbf{R}(x',y') \\ 1 & \text{otherwise} \end{cases}$$

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

$$m(\mathbf{K}(x,y), \mathbf{R}(x',y')) = \begin{cases} 0 & \text{if } |\mathbf{K}(x,y) - \mathbf{R}(x',y')| < \epsilon \\ 1 & \text{otherwise} \end{cases}$$

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, giving the advantage of being more forgiving about the precision of the disparity map. We partition the original key image into a grid of small windows and compute a mean of each window of the grid. Then we warp the pixels of each window of the key image into the reference image and compute the mean 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 key image W_{ij}, the window mean images are computed as:

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

$$\bar{\mathbf{R}}_{ij} = \frac{1}{k^2} \sum_{x',y' \in W'_{ij}} \mathbf{R}(x',y')$$

where W'_{ij} is the set of reference image pixel locations to which the pixels in the key image window were warped. The masking function is defined as:

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

which yields a blocked (or equivalently sub-sampled) background subtracted image. The proposed algorithm is simple and fast enough for real time performance. It is as fast as the normal template based background subtraction algorithm used in [4] in that each pixel is examined once. One obvious advantage is that in using disparity for identifying a candidate background, we can accommodate for changing textures and lighting conditions. The algorithm effectively removes shadows and can be modified to remove surfaces selectively.

2.3 Removing occlusion shadows

The effect of occlusion shadow is illustrated in Figure 2. The problem is that some of the background pixels visible in the key image are occluded by the real object in the reference image. If multiple cameras are available, we can verify the potential object pixels by warping to each of the other reference images and looking for background matches. For a given pixel, "real" objects will fail the disparity test in all instances, whereas the shadow will typically only fail in only one. If there are many objects in a scene it is possible for accidental alignment to cause an occlusion shadow to be shadowed in every reference image, which would then be mistakenly labeled as object. Additional cameras can mitigate this effect.

3 Experimental results

The results of our algorithm are shown in Figure 3. The first pair of images, Figures 3a and 3b show two camera views of an empty scene; Figure 3c is the key image at run time with the person in the view of the camera. The background is subtracted by one of the following methods (Figure 3d-f): 1) simple tolerance range ϵ, 2) simple

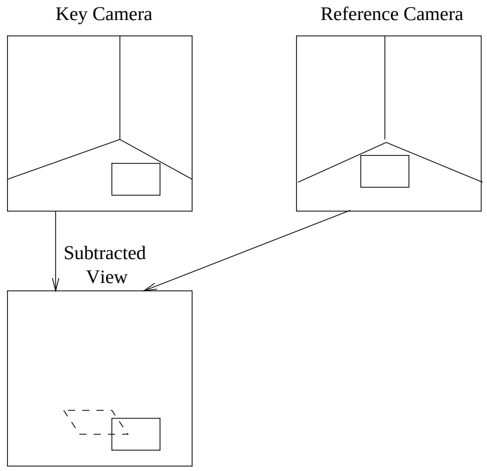

Figure 2: Illustration of the effect of occlusion shadow. Using two camera views in the upper row of the figure we find that there are two areas where the disparity constraint is violated - the "real" match, shown in by the solid rectangle in the bottom figure, and the "shadow" shown by the dotted area. This is the part of the background which is not seen by the Camera 2, while scanning the view of the Camera 1.

tolerance range ϵ on a disparity map, adjusted by the refinement procedure, and 3) windowed means on a refined disparity map. Note the presense of the occlusion shadow right behind the person.

Figure 4 demonstrates the process of removing the occlusion shadow. Three camera views are shown in the top row, where Figure 4a is a left reference camera view, Figure 4b is a key camera view and Figure 4c is a right reference camera view. Figures 4d and 4e show subtraction performed on two pairs of images. Figure 4f shows the result of using all three cameras.

4 Conclusions/Future work

In this paper we described a new approach to background subtraction, which is lighting independent and suitable for real-time processing. 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 is 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. The precision can be increased if some non-linear interpolation of the triangulated mesh is performed instead of the linear one. We also plan to further investigate color calibrating the cameras during the off-line phase of the algorithm. This will help to better determine and reduce the required tolerance ranges for each camera pair.

References

[1] Aaron F. Bobick. Movement, activity, and action: The role of knowledge in the perception of motion. In *Philosophical Transactions Royal Society London B*, 1997.

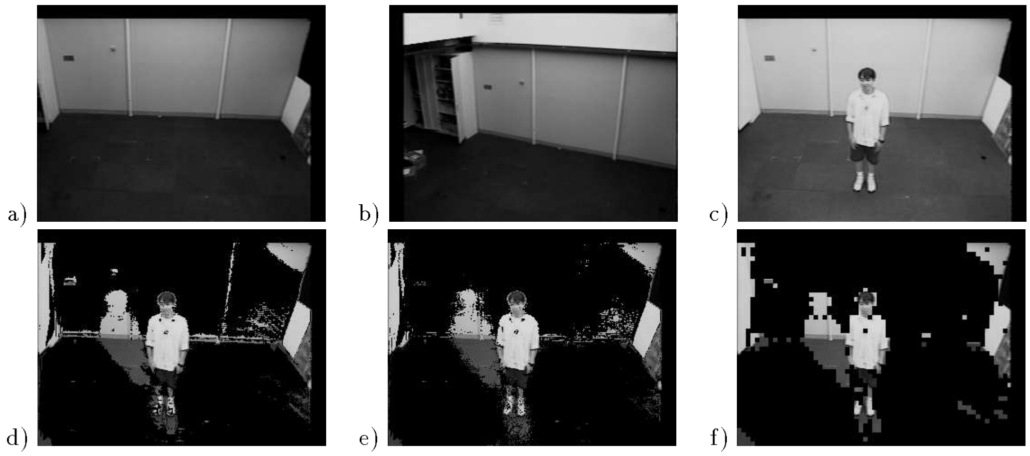

Figure 3: Background subtraction algorithm. (a) and (b) left and right images of the background. (c) Left image with object. Bottom row: results using (d) simple tolerance, (e) simple tolerance on refined disparity map, and (f) windowed means on refined disparity map.

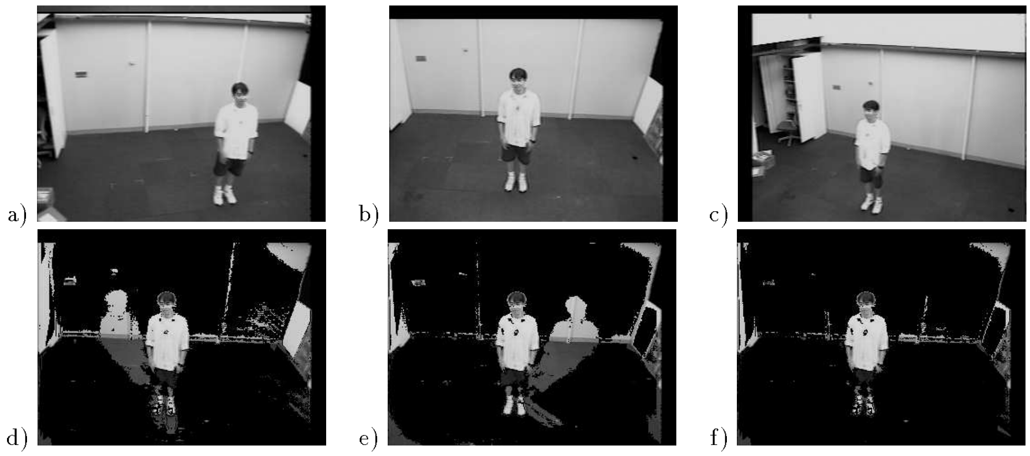

Figure 4: Removing the occlusion shadow. a) left reference camera view, b) key camera view, c) right reference camera view, d) subtraction using a) and b) views, e) subtraction using b) and c) views, f) removing of the shadows using both reference cameras.

[2] Trevor Darrell, Pattie Maes, Bruce Blumberg, and Alex Pentland. A novel environment for situated vision and behavior. In *Proc. of CVPR-94 Workshop for Visual Behaviors*, pages 68–72, Seattle, Washington, June 1994.

[3] James W. Davis and Aaron F. Bobick. The representation and recognition of action using temporal templates. In *Proc. of CVPR-97*, 1997.

[4] Stephen S. Intille, James W Davis, and Aaron F. Bobick. Real-time closed-world tracking. Tr 403, MIT Media Lab, Vision and Modeling Group, 1996.

[5] Ramesh C. Jain and Thomas O. Binford. Dialogue: Ignorance, myopia and naivete in computer vision. In *CVGIP*, volume 53, pages 112–117, January 1991.

[6] T. Kanade. A stereo machine for video-rate dense depth mapping and its new applications. In *Proc. of Image Understanding Workshop*, pages 805–811, Palm Springs, California, February 1995.

[7] M. Okutomi and T. Kanade. A multiple-baseline stereo. In *CMU-CS-TR*, 1990.

[8] Christopher Wren, Ali Azarbayejani, Trevor Darrell, and Alex Pentland. Pfinder: Real-time tracking of the human body. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 19(7):780–785, July 1997.

Appendix A: Incremental constrained Delaunay triangulation algorithm.

Given a set of surface samples, the goal is to build a mesh, that is a faceted (triangulated) representation of the surface. The main challenge here is to find "neighbors" of given a pixel in such a way that the least amount of long and narrow polygons are produced.

1. Initial estimation
   Before the algorithm proceeds it needs an initial estimate of a convex hull containing all the points of the data set. We use the corners of the disparity map as the "initial value" for the triangulation.
2. Including a new point
   A point of the interior area is included in the mesh. The point is connected to its enclosing triangle by three new triangle edges between the point and the vertices of the triangle.
3. Reordering
   The quadrilaterals, which have got the "old" edges of the enclosing triangle as a diagonal, are tested by the maximum angle-sum rule, which states that *in a quadrilateral formed by two adjacent triangles in a Delaunay triangulation, the diagonal goes between two opposite vertices where the sum of the interior angles is greater or equal to* π. This ensures that short and wide triangles are preferred in this triangulation. If the old edges do not meet the criterion, their diagonals are swapped and the new, opposite edges to the inserted point will be examined as diagonals in their quadrilaterals.
4. Iteration
   The Delaunay network now contains one more point. We iterate the two previous steps until all the data points are included.