Real-time detection of independent motion using stereo

Motilal Agrawal, Kurt Konolige and Luca Iocchi
SRI International
333 Ravenswood Ave.
Menlo Park, CA 94025
agrawal, konolige, iocchi@ai.sri.com

Abstract

We describe a system that detects independently moving objects from a mobile platform in real time using a calibrated stereo camera. Interest points are first detected and tracked through the images. These tracks are used to obtain the motion of the platform by using an efficient three-point algorithm in a RANSAC framework for outlier detection. We use a formulation based on disparity space for our inlier computation. In the disparity space, two disparity images of a rigid object are related by a homography that depends on the object's euclidean rigid motion. We use the homography obtained from the camera motion to detect the independently moving objects from the disparity maps obtained by an efficient stereo algorithm. Our system is able to reliably detect the independently moving objects at 16 Hz for a 320 x 240 stereo image sequence using a standard laptop computer.

1. Introduction

Recent advances in computing hardware have led to a rapid decline in computing costs for computing dense depth maps from stereo cameras [9, 15, 1]. As a result, it is now feasible to use real-time stereo systems in a variety of computer vision problems [6, 2, 8, 11]. Augmenting the input space of images with these depth maps results in better performance, robustness, and stability. In this paper, we describe a system that uses dense depth data to determine independently moving objects in real-time from a mobile platform. Visual estimation and detection of motion is a challenging problem in computer vision. The ability to detect, track, and localize moving objects from a mobile platform has applications in automated surveillance and is the first step towards automated video understanding from moving platforms.

Stereo has been used previously to detect and track people from a stationary rig e.g. [2, 6, 3]. For a mobile platform, the first step toward detecting independently moving objects is to estimate the ego motion of the camera. When compared to monocular video, motion estimation from stereo images is relatively easy and tends to be more stable and well behaved [12]. Approaches for binocular motion estimation [13] typically involve establishing feature correspondences. The key steps are to

1. Extract salient feature points in the images.
2. Match feature points between the left and right images of the stereo pair and subsequently triangulate them to obtain $3D$ points.
3. Track these $3D$ points in time and obtain the rigid motion based on these tracked $3D$ points.

In practice, feature correspondences contain few mismatches. Robust estimation methods such as RANSAC [5] are used to take into account the effect of outliers. The use of $3D$ point correspondences to obtain the motion suffers from a major drawback – triangulations are much more uncertain in the depth direction. Therefore, these $3D$ points have non isotropic noise, and a $3D$ alignment between small sets of such $3D$ points gives poor motion estimates. To take into account this anisotropic noise in the $3D$ coordinates, Matei and Meer [10] presented an approach based on a technique from statistics called *bootstrap* to estimate the covariance for the $3D$ points and solve a *heteroscedastic, multivariate errors in variables* regression problem.

An alternative approach is to work directly in the disparity space [4], a projective space with isotropic noise that can be used for efficiently estimating the motion of a calibrated stereo rig. In this paper, we will use a novel combination of triangulation for generating motion estimates and the disparity space homography to evaluate the inliers for the motion as well as to detect the independently moving objects.

We begin by describing the disparity space formulation in Section 2. An overview of our system is given in Section 3. Section 4 describes the feature detection and tracking step which is then use to estimate the egomotion of the

Proceedings of the IEEE Workshop on Motion and Video Computing (WACV/MOTION'05)
0-7695-2271-8/05 $ 20.00 IEEE

camera (Section 5). Section 6 describes the use of disparity space in detection of independently moving objects. Section 7 describes our algorithm for extracting blobs of independently moving objects and tracking these blobs. Implementation details and experimental results on real video sequences are presented in Section 8. Finally, Section 9 concludes this paper with a discussion on ongoing work.

2. Motion in Disparity Space

As in [4], consider a calibrated parallel stereo rig with baseline B. Let f be the focal length and u_0, v_0 be the principal point of each camera. Consider this fixed stereo rig observing a moving rigid $3D$ scene. A point $M \equiv (X, Y, Z)^T$ undergoes a rigid motion with rotation R and translation t so that its new location is $M' \equiv (X', Y', Z')^T$. In homogeneous coordinates,

$$\begin{pmatrix} M' \\ 1 \end{pmatrix} = \begin{pmatrix} R & t \\ 0 & 1 \end{pmatrix} \begin{pmatrix} M \\ 1 \end{pmatrix} \quad (1)$$

The point M projects in the left image to the point (x, y) and its disparity is d. Let $\omega \equiv (x, y, d)^T$ be a point in the disparity space. It is important to note that ω involves image-based quantities only. The errors in these quantities can be considered independent, isotropic, and approximately equal. The noise typically can be assumed to correspond to $\sigma = 1$ pixel. We have

$$\begin{pmatrix} x \\ y \\ d \\ 1 \end{pmatrix} \simeq \begin{pmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 0 & fB \\ 0 & 0 & 1 & 0 \end{pmatrix} \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix} \quad (2)$$

$$\begin{pmatrix} \omega \\ 1 \end{pmatrix} \simeq \Gamma \begin{pmatrix} M \\ 1 \end{pmatrix} \quad (3)$$

Here Γ is the 4×4 matrix on the right side of equation 2 and $\simeq$ indicates that the equality is true up to an overall scale factor. Similary for the point M', we have

$$\begin{pmatrix} \omega' \\ 1 \end{pmatrix} \simeq \Gamma \begin{pmatrix} M' \\ 1 \end{pmatrix} \quad (4)$$

Substituting the inverse of equations 3 and 4 into equation 1, we obtain

$$\begin{pmatrix} \omega' \\ 1 \end{pmatrix} \simeq H(R,t) \begin{pmatrix} \omega \\ 1 \end{pmatrix} \quad (5)$$

$$H(R,t) = \Gamma \begin{pmatrix} R & t \\ 0 & 1 \end{pmatrix} \Gamma^{-1} \quad (6)$$

Thus, in the disparity space, a $3D$ point undergoing a rigid euclidean transformation transforms according to the homography $H(R, t)$. Given the euclidean motion and the camera parameters, it is straightforward to deduce this homography H and vice-versa.

3. System Overview

Our system performs real-time detection of independently moving objects based on tracked feature points in stereo images. It first estimates the rigid motion and the associated disparity space homography of the scene. An efficient real-time stereo algorithm is used to obtain a dense disparity map at each frame. The computed homography is then applied to this dense depth map to synthesize the image at the next frame. This synthesized image is then compared with the actual image at the next frame using sums of absolute differences (SAD) of local windows. Regions with high SAD scores are precisely the independently moving objects since these do not conform to the computed homography. A blob extraction routine groups these regions with high SAD score into blobs and tracks them in time using a Kalman Filter. The key processing steps of our algorithm are

1. Feature detection and tracking
2. Dense depth map extraction
3. Motion estimation
4. Independent motion detection
5. Blob extraction and tracking

4. Feature Detection and Tracking

Harris corners [7] are detected in the left image of each frame in the video sequence. Briefly, the gradients of the input image g_x, g_y are first computed at each pixel. The products of these gradients $g_{xx} = g_x \times g_x$, $g_{yy} = g_y \times g_y$, and $g_{xy} = g_x \times g_y$ are also computed and averaged over a 3×3 window. The corner strength is then computed as $\sigma = g_{xx} + g_{yy} - \sqrt{(g_{xx} - g_{yy}) \times (g_{xx} - g_{yy}) + 4 \times g_{xy} \times g_{xy}}$. This is followed by a non maximal suppression of these corner strengths over a 5×5 window. Finally a threshold on the corner strength is used to filter out corners that are not strong enough. We then interpolate the feature locations to subpixel accuracy. Many of the above steps are efficiently coded using single instruction multiple data (SIMD) instructions.

The features extracted in each frame are matched with the features from the subsequent frame to give feature tracks. Since we have continuous video, a feature point can move only a fixed maximum distance between consecutive frames. For each feature point in the current frame, its SAD is evaluated for every feature point in the next frame that lies within a specified distance of its location in the current frame. Those pixels with SAD scores below a threshold and that achieve mutual minimum SAD are retained and defined as "matched". This makes the scheme more robust as only features that achieve the lowest SAD scores in each

other are reliable. The SAD is implemented efficiently using optimized SIMD instructions.

A dense stereo algorithm [9] is used to obtain the disparity map of each frame. This dense disparity map is needed to detect independently moving objects at a later step and to compute the 3D information of the extracted feature points in each frame.

This processing step results in the extraction of the trajectories of feature points being tracked. For a feature point (x_i^t, y_i^t, d_i^t) in frame t, its corresponding point in frame $t+1$ is given by $(x_i^{t+1}, y_i^{t+1}, d_i^{t+1})$. These feature tracks in disparity space will be used to estimate the motion of the camera.

5. Motion Estimation

We need only three points to estimate the $3D$ motion of the camera. The feature detection and tracking step results in several hundred feature tracks. However, these will have several outliers, arising from two sources. First, the feature tracking itself may result in several false correspondences. Second, these may be points on the independently moving object and hence outliers to the rigid camera motion. Therefore, our motion estimation scheme must be robust enough to estimate the correct motion in spite of these outliers. We use a RANSAC-based scheme to take the outliers into account.

Starting with these potential matches, the RANSAC-based motion estimation involves

1. Hypothesis generation: Three of these corresponding points are selected randomly. The camera motion and the corresponding disparity space homography are obtained from these three correspondences.

2. Hypothesis scoring: The disparity space homography is applied to all the tracks, and the number of inliers to this homography is the score of the hypothesis.

3. Nonlinear minimization: The above two steps are repeated for a fixed number of samples, and the hypothesis with the highest score and the corresponding inliers are taken as input to a nonlinear optimization routine that minimizes pixel reprojection errors of these inlier feature tracks.

We are using RANSAC to compute the dominant motion from the images. This will give us the background motion only if the area occupied by the moving objects is less than the rest of the scene. In other words, if the independent movers occupy most of the image, then we are unlikely to get the background motion through RANSAC.

5.1. Hypothesis generation

Three points are required to generate a motion hypothesis. To get reliable motion, we must ensure that these three points are spaced out well in disparity space as well as in the image. Points that are too close in the image are unlikely to give good estimates of the motion. Therefore, for any selection of three points, we check to see if they are sufficiently spread out in the image.

Next, we triangulate these three points to obtain their $3D$ locations $M_i \equiv (X_i, Y_i, Z_i)^T$ and $M_i' \equiv (X_i', Y_i', Z_i')^T$. We then seek the rotation matrix R and the translation t such that $M_i' = RM_i + t$. This is a standard absolute orientation problem and can be solved efficiently using singular value decomposition [14]. For the sake of completeness, the steps of the algorithm are

1. Compute the centroid $\bar{M}$ and $\bar{M}'$ of the $3D$ point sets $M_{1,2,3}$ and $M'_{1,2,3}$.
2. Subtract the centroid from each point giving us $\hat{M}_{1,2,3}$ and $\hat{M}'_{1,2,3}$.
3. Compute $Q = P'P^T$ where $P = \left[\hat{M}_1\hat{M}_2\hat{M}_3\right]$ and $P' = \left[\hat{M}_1'\hat{M}_2'\hat{M}_3'\right]$.
4. Compute the SVD of Q, given by USV^T.
5. Then $R = VU^T$ and $t = \bar{M}' - R\bar{M}$.

Most of the computations above involve operations on 3×3 matrices and require few flops. The most time-consuming operation here is the computation of SVD of a 3 matrix. We obtain a closed form expression for the SVD of this matrix to save computational time, resulting in very fast implementation.

5.2. Hypothesis scoring

Corresponding to each rotation and translation pair hypothesis (R, t), the disparity space homography $H(R, t)$ can be calculated using equation 6. For a hypothesized correspondence in the disparity space $\omega_i \leftrightarrow \omega_i'$, the homography is applied to ω_i, resulting in the point ω_i". The reprojection error (ε_i) is then given by the difference between ω_i' and ω_i"

$$\begin{pmatrix} \omega_i" \\ 1 \end{pmatrix} \simeq H(R,t) \begin{pmatrix} \omega_i \\ 1 \end{pmatrix} \quad (7)$$

$$\varepsilon_i = |\omega_i' - \omega_i"| \quad (8)$$

A correspondence is taken as an inlier to this homography if the inifinity norm of the error vector $|\varepsilon_i|_\infty$ is less than a predefined maximum threshold value. In our implementation, we have taken this threshold value to be 1.5 pixels. The number of inlier matches to a motion is taken as its

score. Since each of the hypotheses generated during the RANSAC needs to be scored with all the feature correspondences, it is extremely important to code this efficiently. We have coded the routines for equations 7 and 8 using SIMD instructions.

5.3. Nonlinear minimization

The hypothesis with the best score (maximum number of inliers) is used as the starting point for a nonlinear minimization algorithm. We use the Levenberg-Marquardt algorithm for nonlinear least squares minimization. The Jacobian required for this minimization is approximated by forward differencing. Since the 3×3 rotation matrix has only three degrees of freedom, we work with the euler angles instead. The variables for this minimization are therefore the three euler angles for rotation $\Omega \equiv (\Omega_x, \Omega_y, \Omega_z)$ and the three translation parameters (t). The disparity space homography is therefore $H(\Omega, t)$.

For N matches, $\omega_i \leftrightarrow \omega_i'$, $i = 1, \ldots, N$, the error function to be minimized is given by

$$\min \sum_{i=1}^{N} \|\omega_i" - \omega_i'\|^2 \tag{9}$$

where $\omega_i"$ is given by equation 7. The starting point for this nonlinear minimization routine is very good and hence the procedure converges within only 5 to 10 iterations. We have observed that this nonlinear minimization step makes a substantial difference to the computed homography. Motion estimation from just three points is unreliable. The nonlinear minimization step takes into account all the inlier feature tracks and computes more reliable and stable motion estimates.

6. Detecting Independent Motion

The result of the motion estimation step is a homography that specifies the transformation of a point (x, y, d) in the base image to a point (x', y', d') in a subsequent image (the reference image). Conceptually, independent motion detection is simple: just check if the transformed pixel is actually at its estimated location. In practice, motion estimation errors and image noise demand a robust method for this process. We use the following steps:

1. Construct the transformed image by applying the homography to every pixel in the base image for which a stereo disparity exists.
2. Correlate a small region around every pixel in the transformed image with the corresponding region in the reference image, searching over a small area (3×3 window).
3. Accept the pixel as belonging to the rigid background if the correlation value is below a threshold.
4. If not, check for a boundary condition, where the minimum correlation value is less than any correlation around the 3×3 search area. If so, accept the pixel as part of the rigid background.

Consider the two images shown in Figure 1. Figure 1(a) is the base image; from the motion homography and the disparity image, the transformed image is constructed (Figure 1(b)). Notice the holes in the transformed image, where there is no disparity value. There can also be holes at depth boundaries (to the left of the walker), since under certain motions the two sides of the depth boundary will "split" in the next frame, as they move differentially in the image. These areas do not contribute when correlating regions around a pixel.

Given the transformed image and the reference image, each pixel is checked for independent motion by correlation, as described above. Correlation is done with a saturated SAD function to minimize the effect of large differences. If the pixel does not correlate well, according to the checks (3) and (4), then it is declared to belong to an object with independent motion. Note that we perform a small search with the correlation, since it is possible that the motion estimation could transform the pixel to a non integral position; also, there is uncertainty in the motion estimation. As a consequence, a pixel undergoing independent motion must move at least 2 pixels from its projected motion as a non moving object, in order to be detected. Figure 1(c) shows the detected independent motion pixels in white. Note the very low false positive rate, a consequence of the checks (3) and (4).

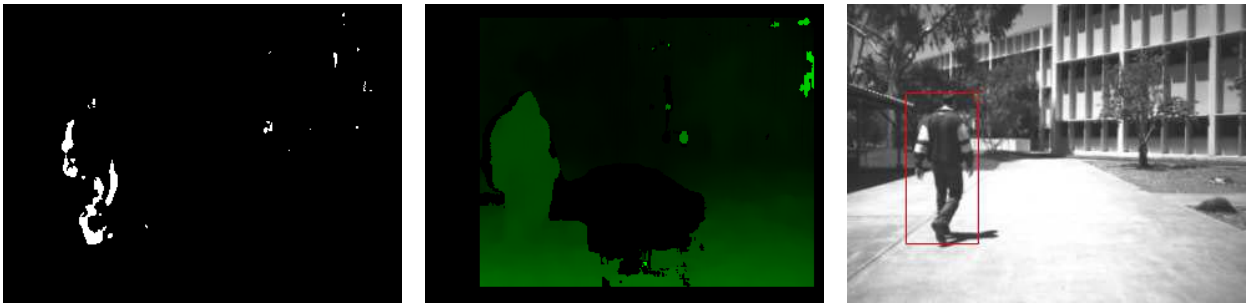
The independent motion algorithm can be implemented very efficiently by using techniques similar to the stereo algorithm [9]. Since we are searching a 5×5 window (3×3, plus all the surrounding pixels), it is roughly equivalent to a search in stereo over 25 disparities. Note that because we are doing detection, we do not have to run an expensive motion search to find where the independent motion pixels have actually moved to.

7. Blob Extraction

The method described so far is able to detect a set of pixels that belong to the independent mover. However, in several applications it is useful to augment the information about the moving objects detected and track them over time. To this end, we have developed a blob extraction and tracking procedure that is able to determine a bounding box of the independently moving object as well as compute its $3D$ position in the world and track it over time. The blob extraction step is based on the assumption that moving objects

(a) Reference image (b) Transformed image (c) Independent motion

Figure 1. Detection of independent moving pixels



(a) Mask from independent motion detection (b) Disparity image (c) Bounding box of extracted blob

Figure 2. Blob extraction results

are surfaces in the world that can be distinguished from the background of the scene; therefore the disparity data associated with the moving objects are somewhat continuous and disjointed from the background (this is a typical case in outdoor scenes).

Given the pixel mask computed by the previous step of independent motion detection, the first step of this processing is to find connected components and compute a set of blobs. At this stage, blobs whose size is below a given threshold are discarded; this accounts for small background blobs that are incorrectly labeled by the independent motion detection routine.

The second step is to group adjacent blobs into *macro-blobs*. The criteria for determining when two blobs should be grouped take into account their distance both in the image space and in the disparity space. Therefore, blobs that are close in the image and in the disparity space are merged into macro-blobs, and each macro-blob identifies a single moving object in the scene.

In some cases, the independent motion detection step can detect only a portion of a moving object (for example, only the legs of a person). In these situations, a third step is necessary for growing the macro-blob into an *extended macro-blob*, which will cover the entire moving object that is being observed. This is achieved by a region-growing procedure that extends the limits of each macro-blob as long as the disparity of the new pixels is compatible with the disparity of the observed object. For example, when objects to be tracked are walking persons, they usually have small disparity variations. Therefore, computing the average and standard deviation of the disparity associated with the motion-detected pixels is sufficient for selecting pixels that are close to the current blob. Figure 2 shows these steps on an example. The result of the independent detection step is shown in Figure 2(a), and the depth map of that frame is shown in Figure 2(b). The final result of our blob extraction routine is shown in Figure 2(c), wherein the missing portions of the person have been filled to get a final blob (shown as a bounding box) that covers the entire person.

The blob extraction procedure is also used to extract the $3D$ location of the moving object. The centroid of the blob and its average disparity are used to get the relative position

of that object. Since we already know the camera motion, we can get the object's absolute position with respect to the initial camera pose. This $3D$ location of the moving object (actually only its projection in the ground plane) is then input to a Kalman filter with a constant velocity model, which retrieves a smooth and consistent trajectory of the moving object. In addition, the filter is used to discard false positives that arise when the independent motion detection step incorrectly labels a consistent blob of background pixels as independent motion for a few frames.

Figure 3 shows the $2D$ track of a tracked object with and without a Kalman filter. The upper plot shows the smooth track of the tracked object as a result of applying the Kalman filter, and the plot below it shows the raw position estimates. Note that the upper plot has been shifted up for better viewing.

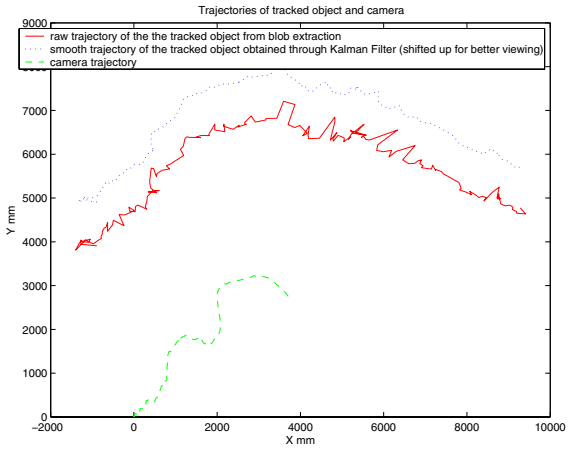


Figure 3. $2D$ trajectories of tracked person and camera

8. Experimental Results

Our system has a few parameters that need to be adjusted based on the particular scenario. In the feature detection step, there is a threshold on the corner strength. This threshold for our experiments is taken as 10.0. We assume that between consecutive frames, a feature can move up to 20 pixels from its previous location. Also, our threshold on the SAD for matching features between frames is taken to be 20.0. The dense stereo algorithm is computed with the assumption of 32 disparities in the scene, and the window size for correlation is taken to be 17×17. We take 1000 random samples for the RANSAC-based motion estimation step. Last, for the independent motion detection step, we have used a threshold of 300 on the SAD score for a 5×5 window.

With these parameters, our system runs at 16 Hz for 320×240 images on a 2 GHz laptop computer with an Intel Pentium 4-M processor. Table 1 shows the detailed timing analysis in milliseconds per frame of the various steps of processing.

Step	Time (ms)
Feature Detection	22
Dense Stereo	7
Feature Matching	7
Motion Estimation	11
Independent Motion detection	12
Blob Extraction	3
Total Time Per Frame	62

Table 1. Timing breakdown

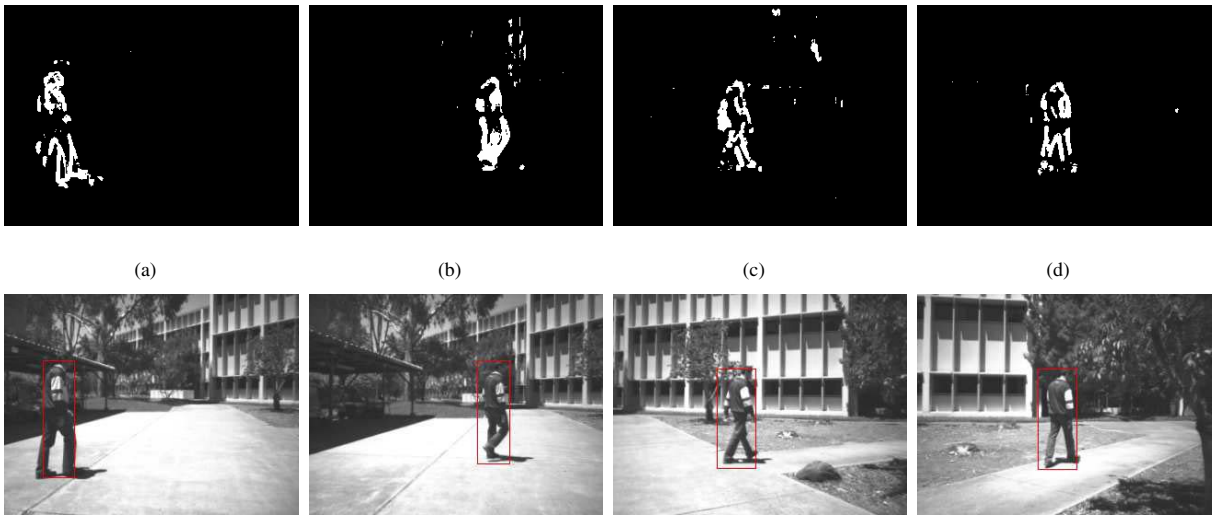
From the timing analysis, it is apparent that a major portion of the time is spent in detecting features. Cutting down time in the feature detection step will result in further improvements in the speed.

Figure 4 shows four frames from our first test video. The first row shows the masks obtained as a result of our independent motion detection routine (Section 6). The bounding box of the extracted blobs for these frames are are shown in the second row. Figure 3 shows the trajectory of the camera as well as the trajectory of the moving person. The entire video showing the extracted independently moving objects is included in the supplemental material (video1.avi).

Figure 5 shows snapshots of the detected objects for another video. This sequence is also included in the supplemental material(video2.avi). Notice that towards the end, our system locks on the shadow of the person instead of the person.

9. Conclusion

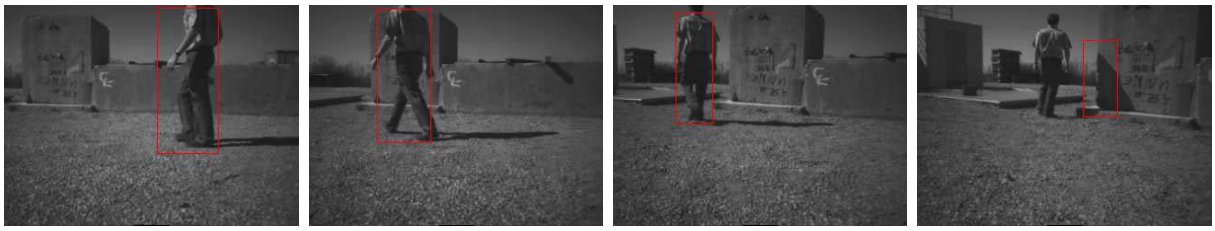
We have described a system that is able to detect, track, and localize independently moving objects in real time from a moving platform. The first step is to compute the ego motion of the camera. This is accomplished in a robust and efficient manner through a RANSAC-based procedure that generates the motion using three randomly selected points. The disparity space homography is then used to find the inliers to this motion. Eventually, the motion with the largest number of inliers is used as a starting point for a nonlinear minimization that minimizes the reprojection errors of these points in the stereo images. This homography is also used



(a) (b) (c) (d)

(e) Frame 25 (f) Frame 75 (g) Frame 125 (h) Frame 175

Figure 4. Results for independent motion detection in video 1



(a) Frame 25 (b) Frame 75 (c) Frame 125 (d) Frame 175

Figure 5. Results for independent motion detection in video 2

to find the independently moving objects from a dense disparity map of the scene. Because we do not have to compute dense optical flow but only verify whether the projected image and the actual image match, our independent motion detection routine is extremely fast. Results from real video sequences are very promising.

Since we do not have ground truth information about the motion of the camera and the independently moving object, we have not performed a detailed quantitative analysis of our results. We hope to accomplish that by comparing against ground truth position estimates from differential GPS. We are also looking into enhancements to make independent motion detection more stable. For example, instead of just comparing adjacent frames for independent motion detection we can skip frames so that the independently moving objects are detected through several frames. This can make the approach more robust to false positives in a few frames in between. This will also be possibly helpful in detecting slow-moving objects with little motion between consecutive frames. In addition, augmenting the tracking with the intensity information of the blob is also likely to result in better tracking.

Acknowledgements

The research was sponsored by the Defense Advance Research Projects Agency (DARPA) under the Department of Interior Award Number NBCH1020014. Any opinions, findings, conclusions or recommendations expressed herein are those of the authors and do not reflect the views of the Department of the Interior/DARPA or Carnegie Mellon.

References

[1] Point grey research inc. http://www.ptgrey.com.

[2] D. Beymer and K. Konolige. Real-time tracking of multiple people using continuous detection. In *Proc. ICCV Frame-rate Workshop*, 1999.

[3] T. Darrell, G. Gordon, M. Harville, and J. Woodfill. Integrated person tracking using stereo, color, and pattern detection. *International Journal of Computer Vision*, 37(2):175–185, June 2000.

[4] D. Demirdjian and T. Darrell. Motion estimation from disparity images. In *Proc. International Conference on Computer Vision*, volume 1, pages 213–218, July 2001.

[5] M. Fischler and R. Bolles. Random sample consensus: a paradigm for model fitting with application to image analysis and automated cartography. *Commun. ACM.*, 24:381–395, 1981.

[6] I. Haritaoglu, D. Harwood, and L. Davis. W4s: A real time system for detecting and tracking people in 2.5d. In *Proc. European Conference in Computer Vision*, pages 877–892, 1998.

[7] C. Harris and M. Stephens. A combined corner and edge detector. In *Alvey Vision Conference*, pages 147–151, 1988.

[8] M. Harville. Stereo person tracking with adaptive plan-view statistical templates. In *Proc. ECCV Workshop on Statistical Methods in Video Processing*, pages 67–72, June 2002.

[9] K. Konolige. Small vision systems: hardware and implementation. In *Eighth International Symposium on Robotics Research*, pages 111–116, 1997.

[10] B. Matei and P. Meer. Optimal rigid motion estimation and performance evaluation with bootstrap. In *Proc. Computer Vision and Pattern Recognition Conference*, pages 339–345, 1999.

[11] L.-P. Morency and R. Gupta. Robust real-time egomotion from stereo images. In *Proc. International Conference on Image Processing*, 2003.

[12] D. Nister, O. Naroditsky, and J. Bergen. Visual odometry. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, June 2004.

[13] C. F. Olson, L. H. Matthies, M. Schoppers, and M. W. Maimone. Robust stereo ego-motion for long distance navigation. In *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, volume 2, pages 453–458, 2000.

[14] S. Umeyama. Least-squares estimation of transformation parameters between two point patterns. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 13(4), April 1991.

[15] R. Yang and M. Pollefeys. Multi-resolution real-time stereo on commodity graphics hardware. In *Proc. Computer Vision and Pattern Recognition Conference*, pages 211–217, 2003.