Proceedings of the 2003 IEEE/RSJ
Intl. Conference on Intelligent Robots and Systems
Las Vegas, Nevada · October 2003

Wide-Baseline Stereo Vision for Mars Rovers

Clark F. Olson[†] Habib Abi-Rached[‡] Ming Ye[§] Jonathan P. Hendrich[†]

[†]University of Washington
Computing and Software Systems
Box 358534
18115 Campus Way NE
Bothell, WA 98011
cfolson@u.washington.edu

[‡]University of Washington
Electrical Engineering
Box 352500
253 EE/CSE Building
Seattle, WA 98195
habib@hitl.washington.edu

[§]Microsoft Corp.
4200 150th Ave. NE
Redmond, WA 98052
mingye@microsoft.com

Abstract

In order to perform localization and navigation over significant distances (up to a few kilometers), it is important to be able to accurately map the terrain to be traversed. Local methods, such as conventional stereo, do not scale well to large distances and prevent long-range planning. In this paper, we discuss wide-baseline stereo techniques for rovers. In wide-baseline stereo, the image pair is captured with the same camera, but at different positions of the rover. While the larger baseline allows improved accuracy for more distant terrain, stereo matching is more difficult for two reasons. First, odometry errors result in uncertain knowledge of the relative positions of the cameras when the images are captured. Second, the change in perspective makes stereo matching difficult, since image landmarks no longer have the same appearance in both images. We address these problems and show test results on real images.

1 Introduction

For the robotic exploration of Mars, a key goal is to maximize the amount of scientific data returned during the fixed span of a mission. This means that a rover must navigate accurately to science targets observed in orbital images or images captured while landing. Since communication with a rover on Mars is usually performed only once per day, the rover must be able to navigate to a target that it cannot see. If the rover fails to reach the goal, an entire day of scientific activity can be lost as the rover again attempts to reach the goal. Current work on rover mapping and localization has partially addressed this issue (for example, [3, 5, 7, 8]).

One issue that has not, yet, been addressed is the computation by a rover of accurate maps of terrain many meters distant. This allows longer range planning and improved localization capabilities. Conventional stereo vision can generate accurate maps for close targets. However, the accuracy of these methods scales poorly for distant targets, since the range error increases with the square of the distance. One solution to this problem is to use a larger baseline (the distance between the cameras), which improves the accuracy of the range estimates. The obvious problem is that a rover with a limited size cannot have two cameras with a large baseline.

We achieve an arbitrarily large baseline using two images captured by the rover at separate positions. While this can improve the accuracy of the range estimation, it introduces new problems. First, stereo algorithms typically calibrate a stereo pair of cameras such that the relative position and orientations of the cameras are known to high precision. This is not possible with wide-baseline stereo, since rover odometry errors prevent such high precision camera positioning. Second, the change in the viewpoint for the images makes stereo matching more difficult, since the terrain no longer has the same appearance in both images.

Our algorithm addresses these problems and consists of the following steps:

1. **Motion refinement.** While our method assumes that an estimate of the camera positions is available, we do not assume that this estimate is accurate. We perform a motion refinement step in order to refine the relative camera position. The overall process consists of the following substeps:

   (a) **Feature selection.** Localizable features are first selected in one of the images using a

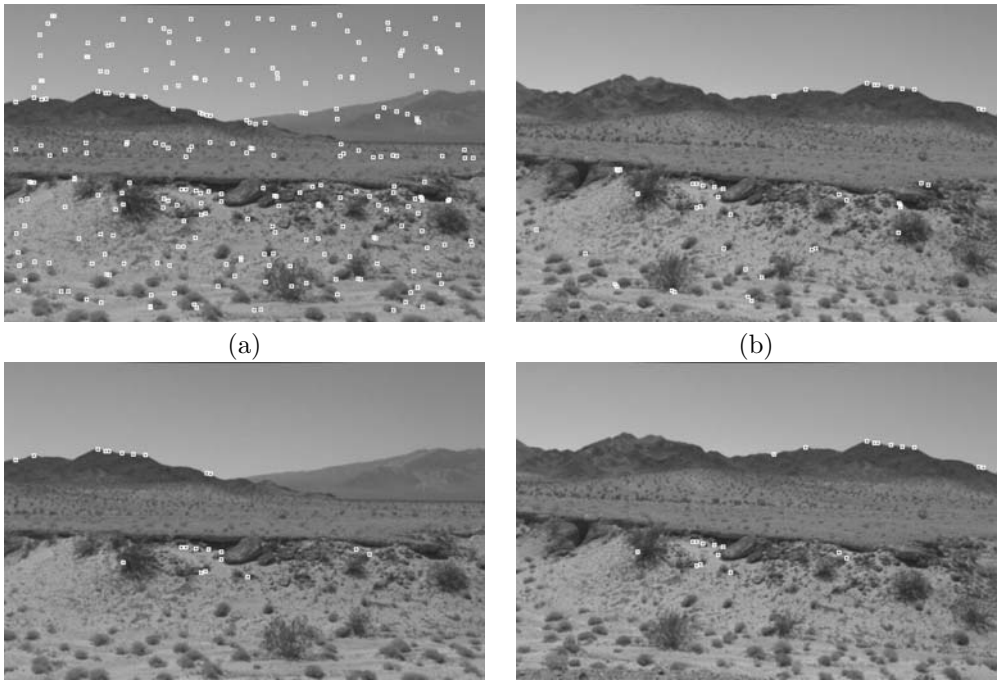


Figure 1: Feature matching example. (a) Features selected in the left image. (b) Initial features matched in right image. (c) Matched features in the left image after outlier rejection. (d) Matched features in the right image after outlier rejection.

variant of the Förstner interest operator [1].

(b) **Feature matching.** Matches for the selected features are detected using a hierarchical search over the entire image.

(c) **Nonlinear motion estimation.** The Levenberg-Marquardt method is used to optimize the motion parameters with respect to the image matches that have been detected. This optimization enforces the epipolar constraints in the stereo image pair.

2. **Image rectification.** In order to facilitate stereo matching, the images are rectified so that the epipolar lines lie along the scanlines of the image [2].

3. **Disparity calculation.** Robust dense matching is performed using maximum-likelihood image matching techniques combined with efficient stereo search techniques. Subpixel disparities are computed by fitting the likelihood surface.

4. **Triangulation.** Once image disparities have been computed, triangulation is performed to determine the three-dimensional position of each of the image pixels.

The remainder of this paper describes each of these steps and shows results using this methodology.

2 Feature selection and matching

In order to refine the motion estimate between the two camera positions, we first detect interest points in one image and find the corresponding matches in the second image. The initial evaluation of each location in the first image is conducted using the Förstner interest operator [1]. This operator scores each pixel based on the strength of the gradients in a region around the pixel. For a pixel to score highly, it must have both strong local gradients in the image and isotropy of the gradient (so that linear edges are discarded.) This is accomplished by examining the covariance matrix of the local gradients. The largest eigenvalue of this matrix is an estimate of the strength of the local gradients and the ratio of the eigenvalues yields the degree of isotropy.

To select individual features, we divide the image into sub-images and select the best local maxima in each sub-image. The reason for the subdivision of the image is to ensure that we select features in each part of the image. A match for each of the selected features

is then sought in the second image.

Matching between the images is performed using a hierarchical multi-resolution search for 7×7 templates centered at each of the selected features. Figure 1 shows an example of the features selected and matched using these techniques. In this case, 256 features were selected in the first image. Of these, candidate matches were found for 42 features in the second image, some of which are incorrect. After outlier rejection 22 correct matches remained for input to the motion refinement step. One direction of future work will be to increase the fraction of selected features that can be robustly matched, for example using the robust matching framework described below.

3 Motion refinement

Motion refinement is the process of adjusting the initial estimate for the robot position and orientation using the stereo matches found in the image. Once the matches have been found as described in the previous section, we apply Levenberg-Marquardt optimization in order to refine the input motion estimate. In particular, we seek precise values for the translation T and rotation R relating the positions of the rover camera at the two locations. This is necessary so that the images can be rectified and the search space for the dense disparity estimation can be reduced to the corresponding scanline. Our optimization method is related to previous nonlinear methods for performing motion estimation from image data (see, for example, [9, 11]).

Our state vector includes not only the six parameters describing the relative camera positions (only five are recoverable, since the problem can be scaled to an arbitrary size), but also the depth estimates to each of the recovered features. With this augmented state vector, the objective function that we are minimizing becomes the sum of squared distances between the detected feature position and the estimated feature position (calculated using backprojection from the current motion estimate):

$$\sum_{i=1}^{N} \left((c_i - \hat{c}_i)^2 + (r_i - \hat{r}_i)^2 \right), \qquad (1)$$

where (r_i, c_i) are the row and column position of the ith feature match in the second image and $(\hat{r}_i, \hat{c}_i)$ are the predicted position of the ith feature in the second image using the current motion and depth estimate. The optimization is iterated, updating the motion and depth estimates at each step, until the objective function is minimized.

The application of these techniques to the images in Figure 1 resulted in an average reprojection error per pixel (the values summed and squared in Eq. 1) of 0.34 pixels. In a series of similar tests, the average reprojection errors 0.31 pixels, with only two outliers having reprojection error greater than 1.0 pixel.

4 Image rectification

In wide-baseline stereo matching, there is often a very large (positive or negative) disparity between matching points in the two images. This necessitates a large search space in the horizontal dimension for the correct match. If a search must also be performed in the vertical dimension, then the overall computation required will be large. We rectify the images using the refined motion estimate so that the matches will be along the corresponding row of the other image (assuming the refined motion estimate is accurate). This is achieved if the images are warped such that they appear as if each camera was rotated to have z-axis is perpendicular to the baseline and x-axis is parallel to the baseline.

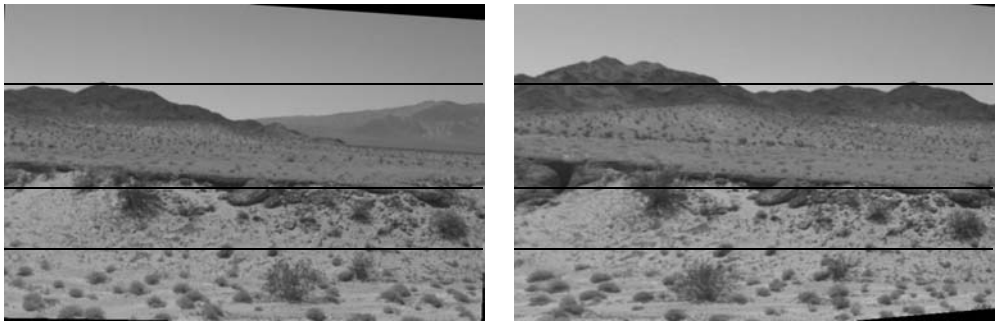
To accomplish the rectification, we use the method of Fusiello *et al.* [2]. Let R_1 and R_2 be the rotation matrices of the two cameras in the world coordinate frame and c_1 and c_2 be their centers of projection in this frame. The camera intrinsic parameters are the same for both images, since they are captured with the same camera. These intrinsic parameters are given by:

$$A = \begin{bmatrix} \alpha_u & \gamma & u_0 \\ 0 & \alpha_v & v_0 \\ 0 & 0 & 1 \end{bmatrix}, \qquad (2)$$

where α_u and α_v are the focal lengths in horizontal and vertical pixels, u_0 and v_0 are the image coordinates of the principal point, and γ is the skew factor (for non-orthogonal axes).

A 3D point with coordinates $[x\ y\ z]^t$ is projected into an image point whose 2D image coordinates in the first image $[u_1\ v_1]^t$ are given by the intersection of the image plane with the line containing the point and c_1. Using homogeneous coordinates, we can write:

$$\begin{bmatrix} u_1 \\ v_1 \\ 1 \end{bmatrix} \simeq A[R_1 \mid -R_1 c_1] \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} = P_1 \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}, \qquad (3)$$



(a) (b)

Figure 2: Images after rectification. Lines have been added to show the relative position of features in the two images. (a) Left image. (b) Right image.

where $\simeq$ indicates proportionality. A similar relationship holds for coordinates in the second image.

The rectifying transformation is a virtual rotation of the left and right camera around their respective centers of projection, with no change in translation. The new rotation R of both cameras in the world coordinate frame must be such that the new image planes are coplanar. This ensures that the epipoles are at infinity, and therefore the epipolar lines are parallel. To enforce horizontal epipolar lines, the baseline must be parallel to the x-axis in both cameras' local frame of reference.

The rotation is constructed of three row vectors as follows:

$$R = \begin{bmatrix} r_1^t \\ r_2^t \\ r_3^t \end{bmatrix} \tag{4}$$

1. r_1 is a unit vector in the direction of $c_1 - c_2$.
2. r_2 is a unit vector orthogonal to r_1 and to the z-axis in the local frame of reference of camera 1.
3. r_3 is a unit vector perpendicular to both r_1 and r_2.

With the new rotation, the projection matrices become:

$$P_i = A[R| - Rc_i] = [Q_i|q_i] \tag{5}$$

and the rectifying transformation is:

$$T_i = Q_i(AR_i)^{-1}. \tag{6}$$

After applying the rectifying transformation, we recenter the images to capture as much of the data as possible within the same image boundaries. Figure 2 shows our sample images after rectifying them according to this process using the motion estimate provided by the previous optimization. The lines drawn in the images indicate that the rectification is good, with corresponding points lying on the same scanline.

5 Disparity estimation

Given the rectified images, disparity estimation is performed by combining a robust template matching method [6] with an efficient stereo search strategy. Our experiments have indicated that the use of a matching measure such as the SSD produces very poor results. We use a maximum-likelihood measure that uses distance measurements from each pixel in the image template to the closest corresponding pixel in the search image. $D_i(\delta)$ is the distance for the ith template pixel at disparity δ. Assuming the distances are independent, our likelihood function is:

$$L(D_1(\delta), ..., D_m(\delta) \mid \delta) = \prod_{i=1}^{m} p(D_i(\delta)), \tag{7}$$

where $p(D_i(\delta))$ is the probability density function (PDF) of $D_i(\delta)$ evaluated at the template disparity δ. In order to find the correct disparity, we find the displacement δ that maximizes the above likelihood function.

Our maximum-likelihood measure improves upon the SSD in two important ways. First, the SSD measure compares only the pixels that are directly overlapping at some disparity of the template with respect to the search image. If camera motion or perspective distortion causes pixels to move by different amounts between the two images, then it will not be possible to find a template position where all of the pixels are correctly overlapped. Our distance measure allows pixels that are not directly overlapping to be matched by linearly combining the distance in the image with the difference in intensity. Computing the best distance for a pixel at a particular template position is no longer trivial with this formulation. However, efficient computation of the distances can be performed by precomputing a three-dimensional distance transform of the input data [6].

(a)

(b)

Figure 3: Computed disparity map. (a) Right image. (b) Disparity image. Dark values represent larger disparities.

The other improvement over SSD matching is that the possibility of an outlier is explicitly represented. In this application, any terrain feature that is visible in one of the images, but not in the other is an outlier for the matching process. Such outliers occur frequently for images taken from different viewpoints. In order to model such outliers, we use a probability density function for each pixel distance that has two terms, one for inliers and one for outliers:

$$p(d) = \alpha p_1(d) + (1 - \alpha)p_2(d). \quad (8)$$

The first term is the error density for inliers and the second term is the error density for outliers, where α is the probability that a particular distance represents an inlier. For inliers, we use a normal distribution in the distance to the closest pixel. For outliers, we use a constant. (This is not a true probability density, but it allows us to model each distance as equally likely in the case of an outlier.)

In order to perform dense matching between the rectified images using the measure described above, we use an efficient search strategy common in stereo vision. This strategy makes use of the observation that a brute-force implementation performs many redundant computations for adjacent positions of the template at the same displacement. We eliminate the redundant computation by storing the information for reuse as necessary for fast matching.

Once the integral disparity estimates have been computed using template matching, we compute a subpixel disparity estimate and an estimate of the standard deviation of the error for each pixel. This is performed by fitting a curve to the likelihood scores near the maximum [5]. Disparity estimates are pruned if the expected error is too large or if the likelihood of the selected location is not large enough.

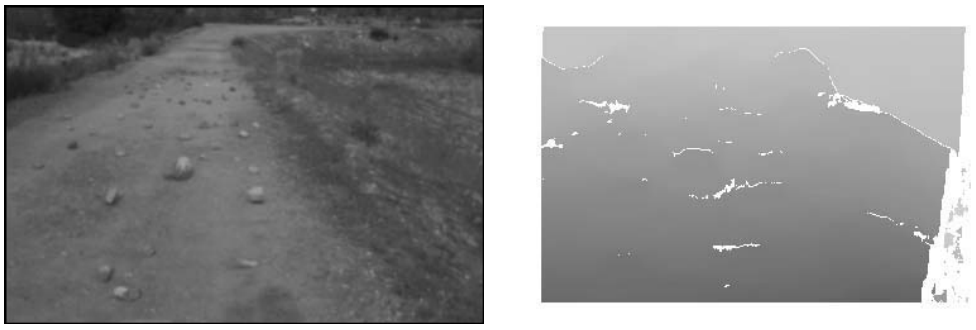
Figure 3 shows the disparity map that was computed for the example in the previous figures. In this case, we obtain sparse results in some portions of the image. This occurs for several reasons. First, there is insufficient texture in thy sky to correctly match the pixels. Second, there are large regions of the image that are not visible in the other image. Note how the high quality disparity matches for the mountains in the background end abruptly about a third of an image width from the right hand side. This is because the mountains to the left of this region are not present in the other image. Finally, the significant change in the viewpoint (approximately 20 degrees) makes matching difficult, particularly in the foreground, where there is very little similarity between the correctly matching features.

Figure 4 shows a second example. Denser results are obtained in this case, since the cameras were pointing in nearly the same directly and the baseline was considerably smaller. Data is correctly pruned from the lower right portion of the image, since this region is not visible in the other image.

6 Rover integration

The wide-baseline stereo method has been integrated with the CLARAty architecture for robotic autonomy [4, 10], which is designed for modularity using object-oriented methods. CLARAty uses generic functional classes that specify the interface and functionality of a generic algorithm. Stereo vision is an example of such a generic functional class. The stereo vision class uses generic image and camera classes in its implementation. In turn, the stereo vision class can be used in the implementation of a visual navigation class.

Our wide-baseline stereo technique has been implemented as specialization of the generic stereo vision class. This allows the wide-baseline stereo techniques



(a) (b)

Figure 4: Computed disparity map. (a) Right image. (b) Disparity image. Dark values represent larger disparities.

to be used in any place that the generic stereo vision class is used in the CLARAty architecture. The current computation time required for the code is around 120 seconds on a 500 MHz workstation. We believe that this can be improved and that a significant computation time is reasonable, since this operation would be performed infrequently.

7 Summary

We have described techniques to allow wide-baseline stereo to be performed on a Mars rover. With these techniques, the rover can accurately map terrain many meters away, unlike conventional stereo. We have overcome two significant problems to achieve good results. Inexact knowledge of the relative positions of the cameras has been addressed using a non-linear motion estimation step. This step automatically selects and matches features in the images in order to refine the initial motion estimate. The problem of robust matching between terrain viewed from different perspectives has been addressed using a robust template matching method that tolerates outliers and perspective distortion. The overall method has achieved good stereo matching results even with a signification change in image viewpoint.

Acknowledgments

We gratefully acknowledge funding of this work by the NASA Mars Technology Program.

References

[1] W. Förstner. A framework for low-level feature extraction. In *Proceedings of the European Conference on Computer Vision*, pages 383–394, 1994.

[2] A. Fusiello, E. Trucco, and A. Verri. A compact algorithm for rectification of stereo pairs. *Machine Vision and Applications*, 12:16–22, 2000.

[3] R. Li, F. Ma, L. H. Matthies, C. F. Olson, and R. E. Arvidson. Localization of Mars rovers using descent and surface-based image data. *Journal of Geophysical Research - Planets*, 2002.

[4] I. A. D. Nesnas, R. Volpe, T. Estlin, H. Das, R. Petras, and D. Mutz. Toward developing reusable software components for robotic applications. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, volume 4, pages 2375–2383, 2001.

[5] C. F. Olson. Probabilistic self-localization for mobile robots. *IEEE Transactions on Robotics and Automation*, 16(1):55–66, February 2000.

[6] C. F. Olson. Maximum-likelihood image matching. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(6):853–857, June 2002.

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

[8] C. F. Olson, L. H. Matthies, Y. Xiong, R. Li, F. Ma, and F. Xu. Multi-resolution mapping using surface, descent, and orbital images. In *Proceedings of the 6th International Symposium on Artificial Intelligence, Robotics and Automation in Space*, 2001.

[9] R. Szeliski and S. B. Kang. Recovering 3d shape and motion from image streams using non-linear least squares. *Journal of Visual Communication and Image Representation*, 5(1):10–28, March 1994.

[10] R. Volpe, I. Nesnas, T. Estlin, D. Mutz, R. Petras, and H. Das. The CLARAty architecture for robotic autonomy. In *Proceedings of the 2001 IEEE Aerospace Conference*, volume 1, pages 121–132, 2001.

[11] Y. Xiong, C. F. Olson, and L. H. Matthies. Computing depth maps from descent imagery. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, volume 1, pages 392–397, 2001.