Visual Target Tracking for Rover-based Planetary Exploration[1,2]

Issa A.D. Nesnas, Max Bajracharya,
Richard Madison
Jet Propulsion Laboratory
California Institute of Technology,
Pasadena, California 91109
818-354-9709
{firstname.lastname}@jpl.nasa.gov

Esfandiar Bandari, Clayton Kunz,
Matthew Deans, Maria Bualat
NASA Ames Research Center
MS269-3
Moffett Field, CA 94035
650-604-4250
{firstname.mi.lastname}@ mail.arc.nasa.gov

Abstract— To command a rover to go to a location of scientific interest on a remote planet, the rover must be capable of reliably tracking the target designated by a scientist from about ten rover lengths away. The rover must maintain lock on the target while traversing rough terrain and avoiding obstacles without the need for communication with Earth. Among the challenges of tracking targets from a rover are the large changes in the appearance and shape of the selected target as the rover approaches it, the limited frame rate at which images can be acquired and processed, and the sudden changes in camera pointing as the rover goes over rocky terrain. We have investigated various techniques for combining 2D and 3D information in order to increase the reliability of visually tracking targets under Mars like conditions. We will present the approaches that we have examined on simulated data and tested onboard the Rocky 8 rover in the JPL Mars Yard and the K9 rover in the ARC Marscape. These techniques include results for 2D trackers, ICP, visual odometry, and 2D/3D trackers.

TABLE OF CONTENTS

1. INTRODUCTION

On the Mars Pathfinder mission, the Sojourner rover used an average of 5.2 Martian days (sols) to drive an average of about 7 meters to a target designated by a scientist and then place an instrument on the target. The Mars Exploration Rovers, Spirit and Opportunity, which will land early in 2004, have a baseline of 3 sols for an approximately 10 m target approach. Future rovers can maximize science return by reducing the number of sols required to approach a designated target, place an instrument, and collect science data. Using fewer sols requires longer traverses between course corrections, which, in turn, requires more accurate navigation. Dead reckoned navigation over rock-strewn terrain can produce navigation errors above the allowable 15% of distance traveled. We have explored the use of visual tracking techniques that combine 2D and 3D information to maintain lock on a designated target point as the rover approaches it. With reliable target tracking, a rover will be able to achieve single-sol instrument placement from 10 to 15 rover lengths away on Mars-like terrain.

The scenario we are exploring is as follows (Figure 1). A scientist designates a target in the imagery downloaded during the previous sol. The designated target location (either the image location or the corresponding 3D location computed from stereo processing) is uploaded to the rover. Once the rover receives the designated target, it will autonomously drive, keeping the target within the rover's field-of-view (FOV), until the target is within the workspace of one of its arms. The rover will then deploy its arm and place the desired instrument on the target for science data acquisition and download to Earth.

Reliable target tracking from on-board rover platforms is particularly challenging for the following reasons:

(i) The visual tracker needs to operate in parallel with a navigator for the rover to safely avoid obstacles when traversing rough terrain. Figure 2 shows Martian terrain that the rover must be able to negotiate.
(ii) A rover experiences sudden changes in its tilt as a result of a wheel dropping off a rock or dipping into a gully, causing the target to leave the camera FOV

[1] 0-7803-8155-6/04/$17.00 2004 IEEE
[2] IEEEAC paper #1392, Version 1, Updated December 10, 2003

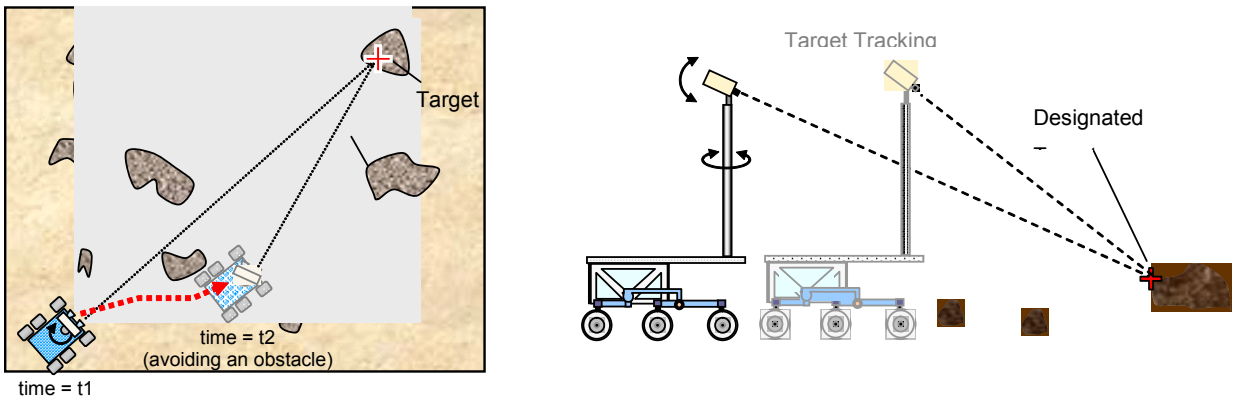

Figure 1: Scenario of rover tracking a designated target

(iii) The target changes appearance both in images as well as in the reconstructed 3D model after it is recovered. High-frame-rate image-acquisition and processing are not feasible due to limited computational and power resources

(iv) The articulated pan/tilt camera system has to meet multiple mission objectives and cannot be fully dedicated to meeting the visual tracking requirements, so image frames may be taken quite far apart in time.

Background

There has been considerable amount of research dedicated to studying the problem of feature tracking. However, most trackers require very small and/or well-estimated motion between consecutive frames or some a priori knowledge of the target geometry as in the case of high-speed tracking of man-made objects in military applications. These assumptions do not hold for a flight rover operating across rough terrain. Researchers at NASA and participating universities have been investigating the adaptations of well-studied tracking techniques to the problem of tracking from a rover platform.

Early work at ARC has investigated the use of binary correlation to register individual features across multiple

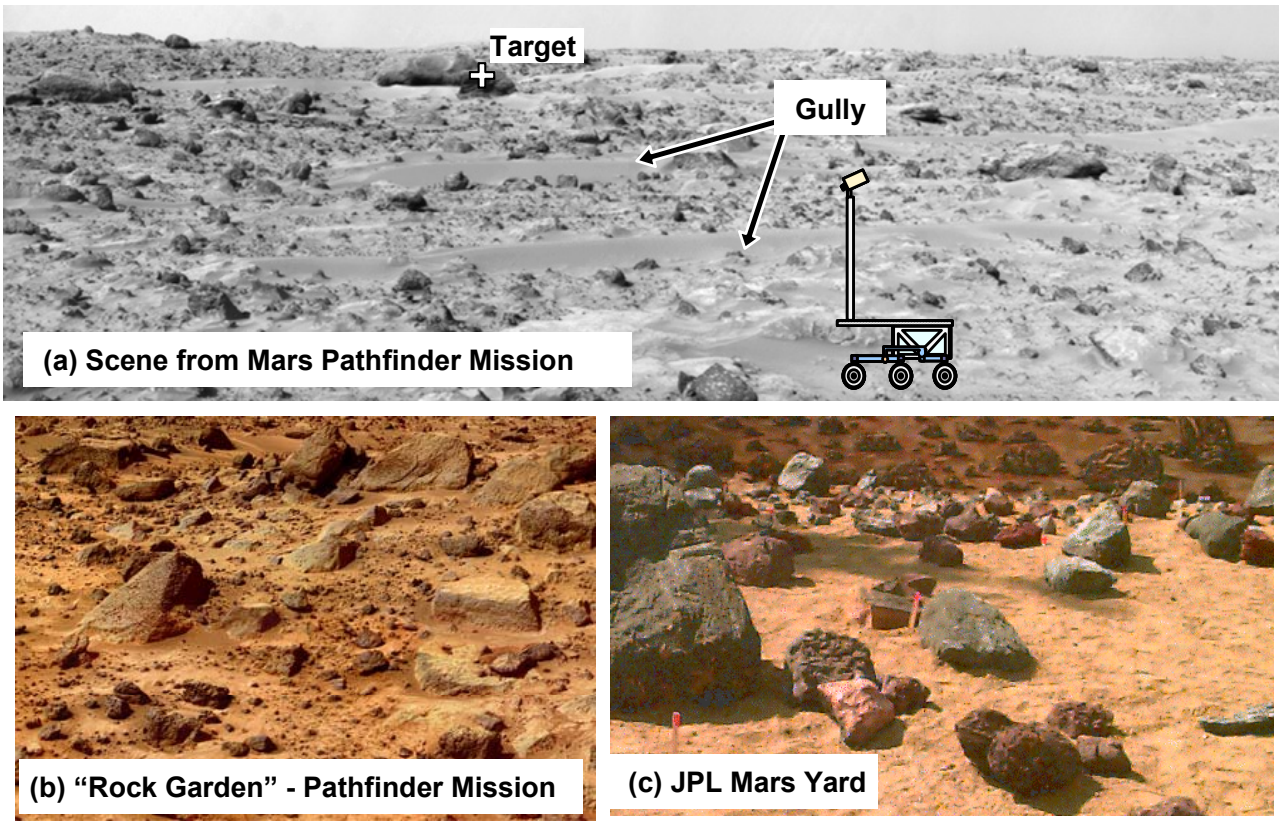

Figure 2: Sample terrain from Mars missions (a) and (b), and the JPL Mars Yard (c)

frames. This binary correlator matches the sign of pseudo Difference of Gaussian (DOG) of filtered imagery. Such pre-filtering has proven very stable in the presence of lighting variations and camera noise. Binary sign correlation was also implemented using logical operation rather than arithmetic calculations, thus increasing the speed while reducing the memory requirements for the tracker. The instruction level parallelism of this correlative approach makes it amenable to the limited processing requirements of near-term rover missions. This technology was demonstrated on the Marsokhod rover for navigating to a desired location [16]. However, as the vehicle approaches the target, the target's image size grows considerably between updates, and a correlation search on the intensity image tends to drift and fail. A recent development at ARC explored the use of surface normals to reproject the 3D terrain model into a virtual depth map, rendering the models as they would be seen from a single range sensor [2]. This approach showed promising results when tested on targets within 5 m from the rover.

Another development in target tracking for rover platforms occurred at JPL. One algorithm, known as visual odometry, uses an interest operator to track intensity-based image features [9][14]. By supplying an initial estimate of the rover motion, this tracking system computes a refined estimate of the rover position and orientation by matching these features in consecutive images. We will present a discussion on how this algorithm is used in our tracker and some of its current limitations. Another algorithm developed at JPL tracked targets in the three-dimensional elevation map that is generated from stereo imagery [11]. This algorithm was developed to enable the autonomous acquisition of targets selected from several meters away. However, changes to the cameras' FOV caused the tracker to lose the designated target.

Another effort at JPL [6] used a homography-based tracking kernel to track designated features. Results for short traverses were promising. Building on these experiences and looking into combining 2D and 3D information to enhance tracking reliability and address the limitations previously encountered, we designed our algorithm to be robust to large changes in feature geometry and photometry between frames and to handle tracking from mast mounted cameras. In this paper, we will first present some of the approaches we examined and why they failed to produce the reliability that we were seeking. We will later show how these initial approaches converge to the current solution whose results are very encouraging.

2. USING ICP FOR ROVER TRACKING

We started by examining how 3D information generated from stereo processing can be used to enhance the reliability of the 2D target trackers. We have explored a number of approaches for using iterative closest point (ICP) algorithms to enhance the reliability of 2D tracking.

Background

The ICP approach [1] is suited for aligning point clouds that represent non-corresponding points on a common surface, which is what we expect from stereo data. Other methods may give better results when the point clouds represent corresponding points. ICP consists of a loop with a two-step kernel. The first step pairs points from one cloud with their nearest neighbors in the other cloud. The second step determines the transform to apply to the points of one cloud to minimize the distance between the nearest neighbors. If, after the transform, some points are not paired with their new, nearest neighbors, then repeating the kernel on new pairings can refine the transform. ICP iterates the kernel until the nearest neighbors do not change, or until iteration produces, for instance, a suitably small average distance between nearest neighbors or a suitably small change in transform.

Most 3D tracking algorithms follow the two steps of the ICP kernel – they find corresponding points in two scenes and then determine the transform that accounts for the motion of the points. The primary difference between the algorithms seems to be in the approach to finding corresponding points.

The ICP approach is suited for point cloud inputs, where it is not necessarily possible to identify "collocated" pixels, or where "gradients" between pixels may not be defined. However, there are other methods for making point correspondences. An example is RANdom SAmpling Consensus (RANSAC) [3], which tests a number of arbitrary pairings of points, finds the best transform to explain each, and keeps the transform that best aligns the two clouds overall.

In this work, we considered two approaches that used the ICP algorithm to recover change in rover pose (position and orientation). The first applies the ICP algorithm to full frames. The second applies ICP to select matches surrounding rock features in the point clouds. In both cases, the algorithm uses a full 6D model of the change in the camera/rover pose. One would expect a more accurate result of either of these approaches as compared to an affine or homography-based feature matcher, which uses an approximate motion model.

ICP on Entire Frames

First, we evaluated the use of full frame ICP between consecutive point clouds. We tested this algorithm on both synthetic images of rocks with known and accurate depth maps generated using a ray tracer as well as on images generated from the rover cameras. We found that ICP converged well when the motion between scenes had a good

initial estimate that major features locked on properly in the first iteration. However, when the algorithm was tested with rover-based images and with estimates of rover motion, the results were not as promising as those of synthetic images. There were several problems that were encountered trying to match terrain point clouds from real images.

The first problem that we encountered with full frame ICP was the convergence of the algorithm on the wrong local minima. ICP is essentially a gradient descent algorithm, and thus it must begin operating in the correct valley of an error surface. Poor initial estimates of the rover motion make it challenging for ICP to converge to the correct minimum.

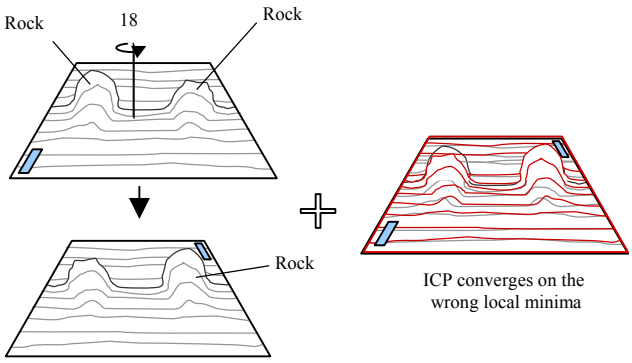

Figure 4: Sources of ICP false matches

To illustrate this point, consider the example of two, differently shaped, squat rocks sitting on a planar patch shown in Figure 4. Suppose two point clouds representing the scene are nearly aligned, so that points on each rock in the first cloud have nearest neighbors on the same rock in the second cloud. Then ICP will pull the neighbors together, improving the alignment of the two scenes. Now rotate the second scene 180° so that the two rocks are falsely matched. Points on each rock now have nearest neighbors on the incorrect rock, but ICP will try to align to them, converging to an exactly incorrect result (see above). Now, rotate the second patch 90° about the same axis so that the rocks in each cloud sit above a planar part of the other cloud. ICP will minimize the distance between nearest neighbors by lowering the first cloud (to move rock points toward the second cloud's plane) and raising it (to move plane points toward the second cloud's rocks), giving no net change. ICP will converge without modifying the initial pose. Finally, separate the two clouds so that they do not overlap. Now ICP may find that all points in one cloud have the same nearest neighbor in the second cloud—the single point closest to the first cloud. ICP will move the centroid of the first cloud onto that single nearest neighbor, and apply an arbitrary rotation, potentially producing any of the situations described above. In summary, ICP must begin with a reasonable estimate to converge properly. It follows that ICP must supplement another tracking method, such as 2D tracking or visual odometry.

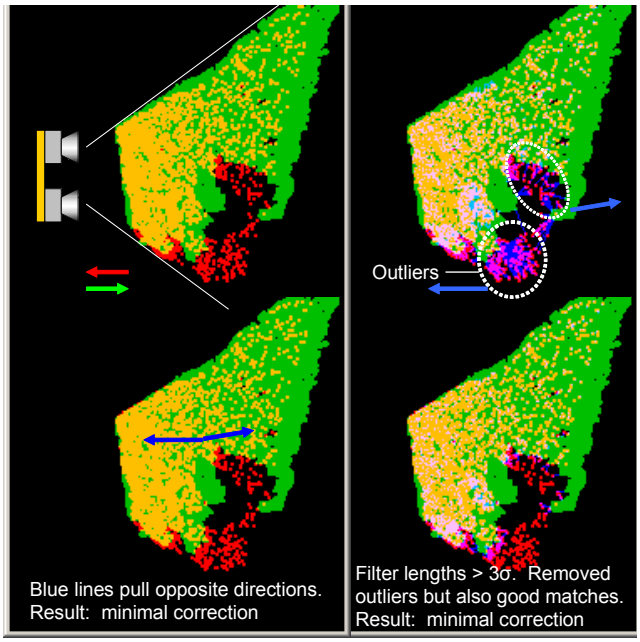

Figure 3: Top left shows the top view of two point clouds computed from stereo processing acquired at time t (green) and t+dt (red). Top right shows that areas of missing data result in selected neighboring points that pull point cloud in opposite directions. Bottom left shows overall resultant change to match two clouds is very small. Bottom right shows that even after filtering outlier regions, the overall result does not cause the two point clouds to match up

The second problem stemmed from stereo noise, which was present in point clouds generated by applying stereo processing [9] to either synthetic or rover-based images. Stereo contributed at least two error sources that hampered ICP.

First, stereo-generated depth maps have fattened occluding edges. Stereo finds the depth at an image pixel by correlating a window around the pixel, across a second image, and converting the offset ("disparity") that produces the best correlation value into a distance from the cameras. A window containing an occluding boundary can correlate well at the disparity of the foreground object or the background, and favors the more distinct foreground even when the window is centered on the background. As a result, several pixels outside an occluding boundary are incorrectly assigned the distance of the foreground object. When the scene rotates, the artificial edges stay with the foreground object, but they do not rotate. ICP then underestimates rotation as it averages the rotating and non-rotating pixels.

Second, stereo-generated point clouds have "outlier" regions, which appear in only one of two consecutive point clouds that ICP must align. As any range sensor moves around the scene, different parts of the scene become occluded. The problem is aggravated for stereo as we attempt to remove fluke results. Correlation-based stereo

can produce occasional fluke results, given occasional hard--to-track features plus image noise. Solutions generally involve eliminating or modifying pixels whose depths are inconsistent with their neighbors, or small islands surrounded by unrecoverable depths. If stereo cannot find a dense map, eliminating disconnected, small patches can remove coverage for large areas. Figure 3 (top left) shows two point clouds (red and green), where a large area along the bottom of the red cloud has been eliminated as described here. ICP must realize that the region is an outlier, and then eliminate it. Such outlier rejection is typically done by eliminating pixels that are too far from their nearest neighbor. However, as the picture shows, the outlier region is as close to the green cloud as is another, valid, red region. Eliminating both regions leaves nothing for ICP to align, but retaining them causes ICP to give up, unable to align both regions. In other instances, where regions did not compete, outlier regions often caused ICP to iterate to a poor transform before the outliers could be eliminated, and ICP could rarely recover.

In summary, full frame ICP suffered from several problems. It had to begin with a reasonably good estimated transform in order to converge to the correct local minimum. However, even with a perfect estimate, it was inclined to diverge to account for large outlier regions and illusory points on occluding boundaries. It should be possible to eliminate both of the problems with stereo processing, at which point ICP may deserve a second look.

ICP on Select Points

To avoid problems with full-frame ICP, we attempted to use ICP to align small patches in one cloud with the entire second cloud. Small patches being tracked into a second image necessarily appear in the first image, and if they do not appear in the second image, ICP-confusion would be acceptable. Small patches also might be chosen to avoid occluding edges. In addition, small patches may avoid another ICP difficulty—aligning sparse rocks on a flat terrain, where the good match between flat terrain in two clouds masks poor alignment of the sparse rocks.

Once again, the results of this experiment on real images were not promising. Because of the limited size of the patch surrounding the feature, and because of an imprecise initial pose estimate, the patch may not have significant overlap with its proper mate in the second cloud. As previously discussed, this can produce an arbitrary transform. Second, because of the small size of the patch, ICP becomes more sensitive to stereo noise in the patch pixels. Third, because of the small size, it is more difficult to distinguish rotation from translation. As with full image stereo, the latter problems can be reduced, in this case by treating several small patches as a single cloud, thus adding a "lever arm." However, the need for rather accurate initial alignment leaves ICP to be used as a post-processing step, and increasing the amount of image coverage increases the probability of including outlier regions.

Conclusion

Initially we considered using the Iterative Closest Point matching algorithm (ICP) to aid the 2D/3D tracker to improve the tracking accuracy. Stereo is sufficiently noisy that it produces entire blobs of data that appear in only one image of a pair. These areas must be rejected as outliers before ICP can converge. However, when the images are not initially well-aligned, these outliers look like any other poorly aligned area. With no easy way to eliminate the outliers, ICP will likely not converge to any credible answer Because ICP requires both good initial estimate and good stereo data with low noise, only select points with high stereo confidence (*i.e.* small covariances) can be used effectively. This degenerates to an algorithm that tracks sparse 3D points that have high stereo confidence, and uses them along with the rigid body assumption to eliminate drifting points and track an obscured point. This algorithm is known as visual odometry [9][14].

3. VISUAL ODOMETRY

The JPL Machine Vision group currently uses an implementation of visual odometry based on Matthies' dissertation [9], with several subsequent improvements to speed and accuracy. This algorithm is the current visual odometry pose estimation baseline in CLARAty (Coupled Layer Architecture for Robotic Autonomy) [12]. CLARAty is an architecture for reusable robotic software which integrates a number of robotics and vision technologies into its framework. The current implementation of visual odometry is able to recover translation of the imager with about 2% error of the total traversed distance [14], and rotation with about 3% error of the heading change.

The current visual odometry implementation works as follows. Start with a pair of images from a pair of cameras with known relative geometry—a stereo head. Choose distinctive features in, say, the left image, stereo-match them into the right image, and triangulate to get the 3D positions of the features relative to the cameras. Move the camera pair and acquire a new image pair. Track the features from the old left image into the new left image. Stereo match the features from there into the new right image, and triangulate to find the 3D feature positions relative to the new imager locations. Compare 3D distances between features in the first frame with the distances between corresponding features in the new frame, and eliminate features that move with respect to their neighbors. Using a rigid world assumption, use a maximum likelihood estimator to determine the 6D pose change of the imager that best accounts for the apparent 3D motion of the features [9].

Using visual odometry alone to estimate target location results in accumulated errors that are too large for accurate

instrument placement. The pose estimation errors from visual odometry are cumulative and will lead to larger errors in tracking accuracy compared to algorithms that use designated target information and control the camera gaze for that target. Hence we use the visual odometry as an initial estimate of the rover change in pose to seed a tracker that focuses on the designated target.

While the visual odometry algorithm produces promising results for rover pose estimation, this algorithm can lose track of the target if the change in rover pose is large between consecutive frames. This occurs when a rover falls off a rock during a traverse.

Filtering of non-flat features

Visual odometry tracks a large number of corner feature using a homography transform. The tracker assumes that the corner features fall on flat surfaces and not on occluding boundaries. While outliers are filtered using a Maximum likelihood estimator after being tracked, we explored whether selecting fewer better features would improve the robustness and the pose estimate computed by visual odometry. Since trackers require flat features for tracking, we applied a surface normal filter (Figure 5) that removes features that are not flat.

Experimentally, normal filtering eliminates poor features, but reducing the number of features reduces the probability that visual odometry will succeed. Any improvement in quality of tracking is masked by the errors that accumulate on frames where visual odometry fails.

We came to this conclusion by testing visual odometry on a rover commanded to traverse an S trajectory of about 10 m using several forms of normal filtering (Table 1). We ran this algorithm on imagery acquired from the hazard cameras (body cameras). We collected data for four cases:

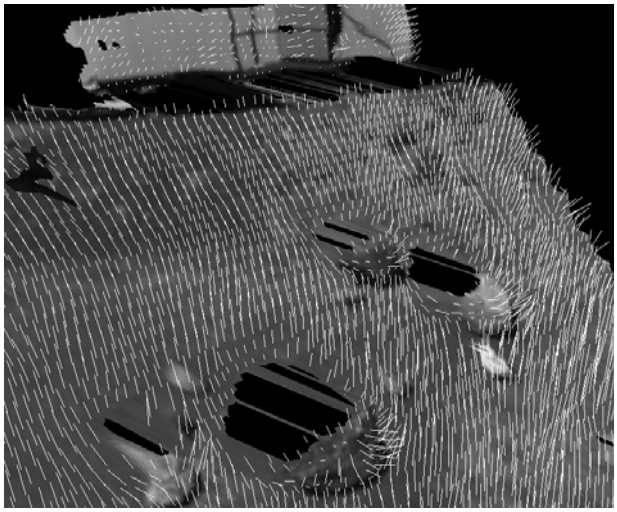

Figure 5: Surface normals on a sample terrain

1. **Baseline:** is the CLARAty visual odometry. We acquired 200 features and typically lost around 80 during tracking.
2. **85%**: we generated a depth map of the scene using JPL stereo, converted to a normal map, and recorded the confidence associated with each normal. We eliminated any features with normal confidence less than 85%, as these represent non-flat features that do not meet assumptions in visual odometry's homography transform. That typically eliminated about 10 of the 200 features.
3. **Best-15:** we acquired 200 features, and then chose the 15 with the highest normal "confidence." These should be the flattest features, most suitable for homography transform, and thus most likely to track properly in visual odometry. In general, few of these features were lost in tracking.
4. **Best-30:** analogous to Best-15, but with 30 features.

We recorded visual odometry after approximately 3 m, 5 m,

Table 1: Results of using surface normal filter on visual odometry features

Test Run	Length	x (m)	y (m)	z(m)	% Total Error	xr (rad)	xr (rad)	zr (rad)
Ground Truth	3 m	2.793	0.390	0.015		0.018	-0.094	1.115
	5 m	5.834	2.385	0.017		0.016	-0.149	0.017
	10 m	9.152	1.631	0.113		-0.015	0.012	-0.325
Baseline	3 m	2.808	0.415	0.044	1.46%	0.038	-0.084	1.105
	5 m	5.935	2.433	0.046	1.83%	0.214	-0.159	0.479
	10 m	9.360	1.813	-0.205	4.53%	0.016	0.115	-0.290
85%	3 m	2.805	0.409	0.040	1.19%	0.039	-0.085	1.105
	5 m	6.083	2.157	0.246	6.47%	0.213	-0.222	0.442
	10 m	9.470	1.449	-0.057	4.34%	0.017	0.258	-0.332
Best-15	3 m	2.816	0.404	0.200	6.63%	0.013	-0.162	1.096
	5 m	7.028	0.95	0	**29.6%**	0	0	.436
	10 m	10.28	0.288	-0.128	**19.0%**	-0.006	0.453	-0.306
Best-30	3 m	2.800	0.408	0.045	1.27%	0.040	-0.087	1.102
	5 m	6.672	1.069	0.387	**25.4%**	0.390	-0.138	0.286
	10 m	9.984	0.233	-0.405	**18.4%**	0.063	0.448	-0.332

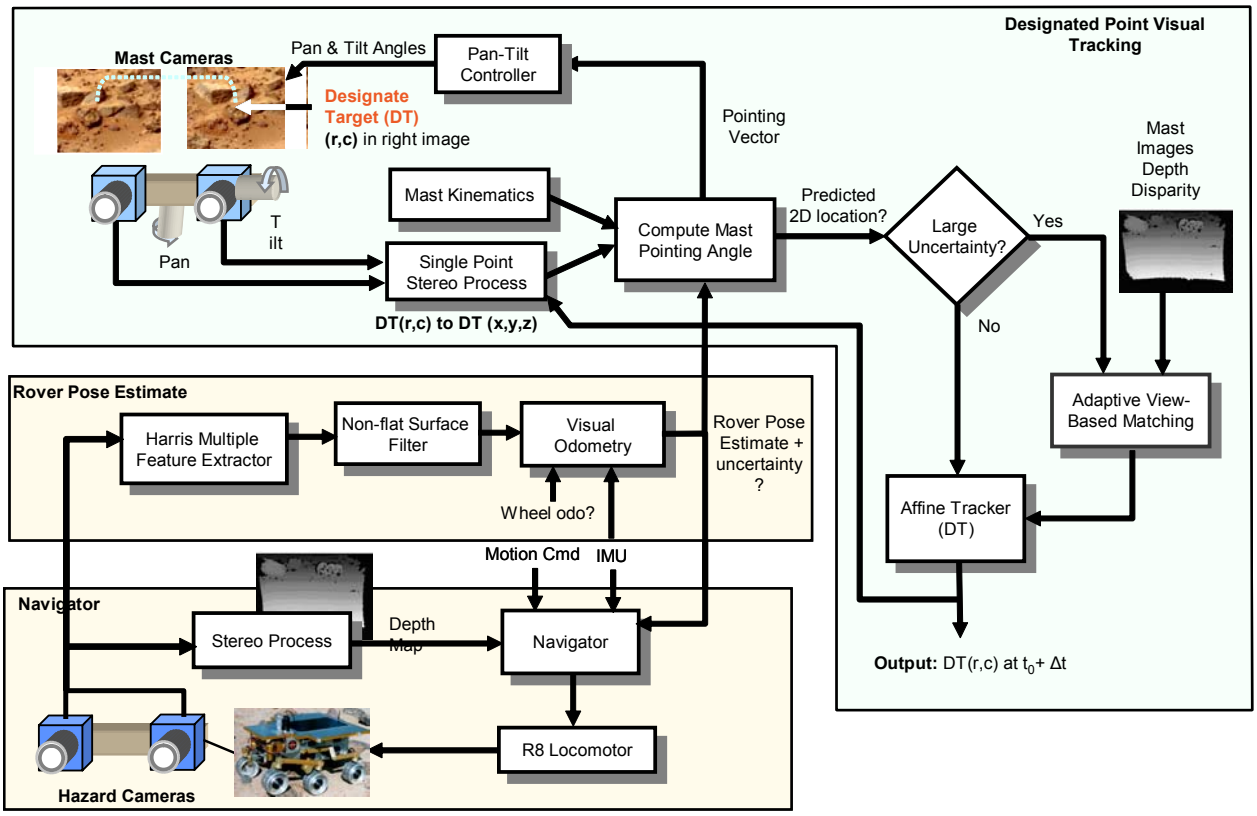

Figure 6: Overall Visual Tracker System Architecture

and 10 m of commanded traverse. Table 1 presents recovered translation in meters and rotation vector in radians.

An important feature of the results above is that they depend significantly on the frequency with which visual odometry fails. When visual odometry fails to track at least 8 features, it reports zero for zr, xr, and yr. This happened in the frame at 5 m for the **best-15** case. The **baseline** and **85%** cases had 4 frames that failed to track. The resulting error in zr, xr, and yr is the likely cause of the poor recovery in those dimensions. The **best-15** case had 26 failures, and **best-30** had 13 failures. The increasing number of failures is clearly reflected in the reported error.

To restate the conclusion, reducing the number of features, even the less accurate features, by normal filtering reduces the probability of successfully tracking 8 features, which increases the probability of a jump in accumulated visual odometry error. It is safer to use more, lower quality features.

4. THE 2D/3D VISUAL TRACKER

Building from these results, we developed a visual tracker that controls an articulated mast to keep a designated feature in the camera FOV while driving towards the target. Figure 6 shows the visual tracker system architecture. Initially, a scientist selects a point in one of the stereo images acquired from the mast (top left). The stereo triangulation algorithm computes a 3D location for the target. As the rover moves, its body-mounted cameras feed images to a visual odometry algorithm, which tracks 2D corner features and computes their old and new 3D locations. The algorithm rejects points whose 3D motion is inconsistent with a rigid world constraint, and then computes the apparent change in rover pose. Using this estimated change in rover pose, the 3D position of the target feature, and the mast kinematics model, we compute the pan and tilt angles of the mast to keep the target in the center of the camera's view, minimizing the area over which the 2D tracker must operate. If the motion between consecutive frames is still large (i.e., 2D tracking was unsuccessful), the tracker applies an adaptive view-based matching technique (next section) to the new image. This technique uses correlation-based template matching where it scales the feature template based on the depth ratio between the original template and pixels in the new image. This is repeated over the entire search window and the best correlation results indicate the appropriate match.

The simplest method of selecting and approaching a target for instrument placement is to acquire a stereo image pair from the mast, compute the 3D location of the target, and then drive to the target using rover pose estimation. Unfortunately, even very small errors in stereo ranging or rover pose estimation translate to a very large error in the prediction of the target location after a 10 m traverse. The

large initial distance makes stereo range accuracy sensitive (a 0.2 pixel disparity error of our 4 mm 640x480 cameras with a 9.9 µm pixel size and 19 cm baseline corresponds to a 25.4 cm range error) and good pose estimation prone to accumulated error (1% error would accumulate to 10 cm over 10 m). After approaching 9 m toward a target 10 m away from a mast 1 m high, the error in the horizontal and vertical positions of the target is approximately equal to the range error at the starting point. So if the range (or pose estimation) error is 10 cm from 10 m away, the target position error is approximately 10 cm when the rover is at 1 m from the target.

Tracking the target to even within several pixels after a 10 m traverse improves the estimated target position considerably over even good stereo and pose estimation. Using 4 mm cameras on the mast of the rover, tracking can achieve better than 2.5 cm accuracy: a single pixel at 10 m corresponds to 2.5 cm; at 1 m 2.5 cm corresponds to 10 pixels; in general we only see several pixels error after tracking over 30–40 iterations of 25 cm traverse per iteration. Even more accurate tracking can be done with the 16 mm cameras. However, because at 10 m, a 1.7 degree pointing error corresponds to 100 pixels error (compared to a 9.4 degree pointing error for the 4mm cameras), the mast must be pointed much more accurately to maintain a fix on the target. Even with a large search window, the mast must be pointed to within several degrees of accuracy.

Rover Pose Estimation

In order to achieve the accuracy needed to point the narrow (16 mm) FOV cameras, we use an IMU to determine the vehicle's roll and pitch and visual odometry [14] to determine the vehicle's yaw and 3D position. Although visual odometry is prone to accumulated error, we are only interested in the relative accuracy, which should be well within 1° pointing and 1% of the distance traveled between each iteration of the algorithm (25 cm, resulting in 0.25 cm position error).

2D Tracking

The tracking algorithm uses (only partially optimized) normalized cross correlation [7] between each frame to seed an affine matcher [8][15] that maintains a single original template and simply updates its affine parameters on each frame. The fast cross-correlation matcher allows for very large motions of the feature window between frames while the affine matcher provides accurate localization and prevents accumulated error (drift) by using a single key frame. Depending on the motion expected, the affine matcher can optimize on the feature translation and either all affine parameters, only scale parameters, or only in-plane rotation parameters.

Experimental Results

Figure 7: The Rocky 8 rover with the mast unstowed

For our experiments we used the Rocky 8 Mars rover prototype (Figure 7), which has an articulated mast with two stereo camera pairs: navigation and panoramic stereo pairs (Table 2). These are used for the visual tracker. The wide FOV, body-mounted cameras are used for visual odometry. In this example, the rover motion commands are simulated to mimick the onboard navigation algorithm. These commands consisted of arcs ranging from 1-3m with a heading change of up to 45 degrees. Images from two stereo pairs (4 mm and 16 mm camera pairs) were taken every 25 cm from a mast raised 1m off the top of the rover (approximately 1.5 m off the ground) and pointed at the target using the estimated rover pose and mast kinematics. Additionally, images from the front body cameras (hazard cameras) where taken and used for the visual odometry algorithm. Accelerometers from an onboard IMU were used to determine pitch and roll of the vehicle (while stopped) every time images were taken. The rover was driven over the most severe conditions it was designed to traverse (over rocks about 30 cm high), resulting in significant wheel slip and changes in roll and pitch.

Table 2: Camera Configuration for Rocky 8

Camera Name	Placed On	Baseline	Lens	FOV	CCD Resolution	Pixel size (µm)
Navigation	Mast	19 cm	4 mm	60°	640x480	9.9
Panoramic	Mast	23 cm	16 mm	17°	1024x768	4.65
Hazard	Body	8.6 cm	2.8 mm	90°	640x480	9.9

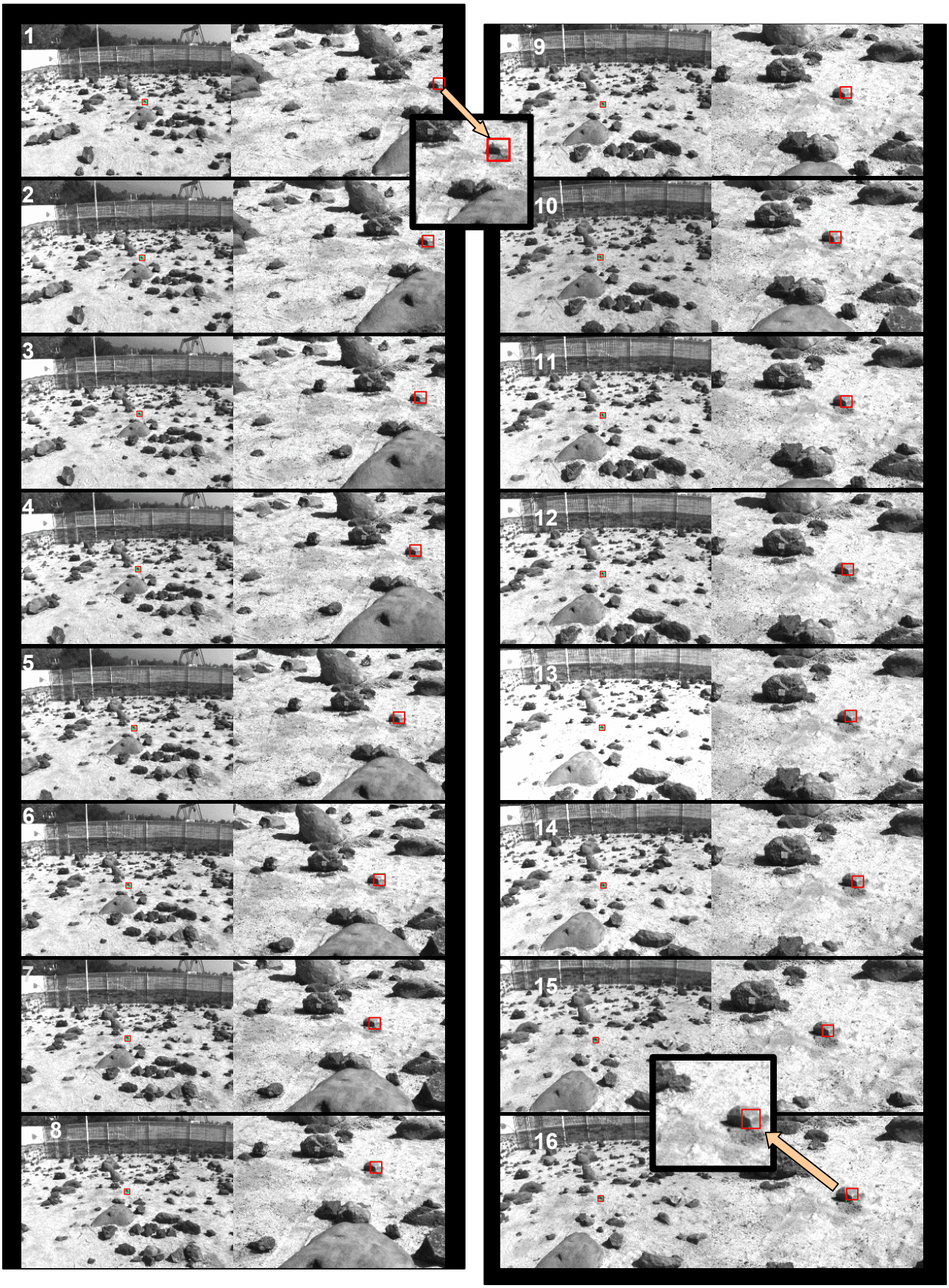

Figure 8: Data from a 10 meter run tracking with both Pan and Nav cameras

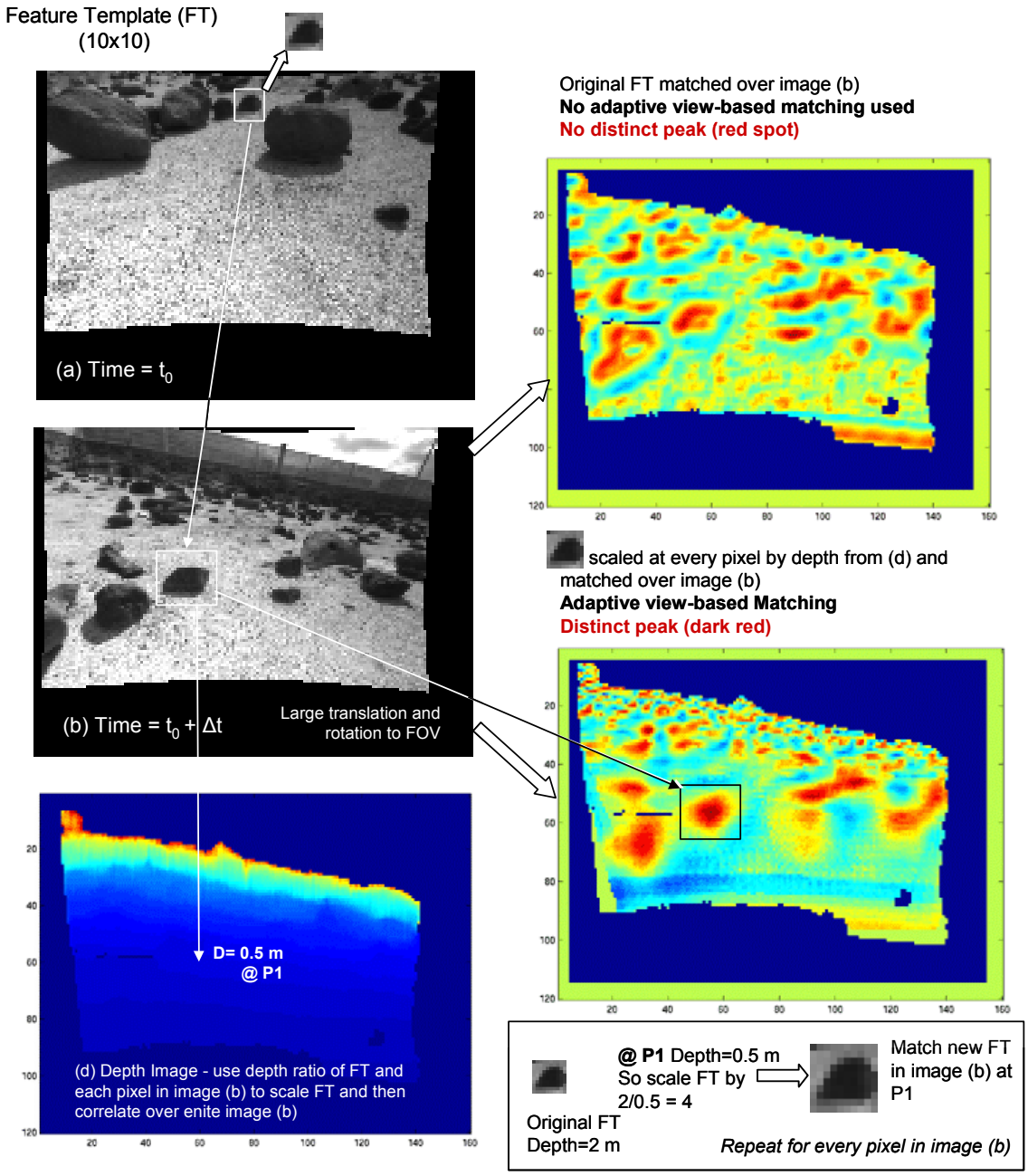

Figure 9: An example of adaptive view-based matching with correlation results shown on the right

Preliminary results of this algorithm (Figure 8) showed that targets were tracked through an 8 meter traverse under realistic Mars-like conditions. Several runs were completed and designated targets were successfully tracked within a pixel of the selected feature point. The implementation of the adaptive view-based matching is complete but being integrated with the current tracker to compensate for larger changes in motion.

5. Adaptive View-based Matching

When the rover drops off a rock, the mast cameras, especially the narrower FOV lenses experience large and sudden changes. Under these conditions, even the hazard cameras experience enough change that causes the visual odometry algorithm to fail. In the absence of a good initial estimate, the tracker has to search a large window in order to recover the correct pointing direction and do a finer search on the target using the narrower FOV cameras.

Correlating a template over an image may be an effective way to recover a feature being tracked if the feature has a unique appearance and its appearance has not changed significantly. However, correlation can be improved by using a template transformed with a scale and orientation adaptively selected based on the 3D information. This approach uses the notion introduced by Olson [13], and is similar to one utilized by Morency [10]. Instead of correlating a fixed template over a new image, the template is reprojected according to the depth and surface normal information at each point in the correlation. Essentially, the template is being shifted over the 3D surface of the terrain in order to register a match.

Figure 9 (left) shows images taken in sequence after the rover has traversed some distance and driven its right side over a rock. The feature template representing the selected target is scaled at every pixel in the new image based on the depth information from image (b) at that pixel. Correlation results are shown on the right hand side of Figure 9, where the top image shows the results without appropriate depth scaling and the bottom image with the proper depth scaling. The latter image shows a region with a very dark red sport indicating the highest correlation results and the proper match for the original feature template.

To reduce computational cost, the template is assumed to be planar, in which case a planar transformation can be applied to the template at each correlation point. Without this planar assumption, a mesh will be needed for the feature, which would then be transformed in 3D based on information acquired from stereo processing.

Because the adaptive technique relies mainly on relative depths, it should not be particularly sensitive to the stereo depth accuracy (as well as camera model accuracy). Furthermore, because the adaptive technique relies on the ratio of depths, it should be less sensitive to depth errors at far distances typical of stereo matching [18]. In addition to facilitating the recovery of a lost target, we plan to use an adaptive scale method to track a template over a longer distance and consequently reduce the accumulation of drift when tracking.

While the adaptive correlation technique is more expensive than a conventional one, it can be done at a low resolution to simply find an initial match, and then refined at higher resolution with a more accurate matching technique that requires a good initial guess (such as an affine matcher) [8][15].

The approach still cannot account for one degree of freedom: the rotation around the optical axis (camera roll). However, the rotation will be recovered from the IMU's accelerometers, which measure the vehicle's tilt. This information will seed an algorithm that will match far-field features to refine the vehicle roll estimate. With stereo data, features that are far away are easy to detect, and matching is easier because features in consecutive images will be approximately the same scale.

Initially, off-line tests of scenarios where large changes in viewing directions occur yielded very promising results using depth information only to scale the template. Future work will integrate this with the visual tracker system.

6. SURFACE NORMALS FOR POSE ESTIMATION

The computing of surface normals from 3D point clouds that we visited earlier can also be used to recover rover pose. Surface normals for corner features that are tracked can be used between consecutive steps to recover the change in rover pose. This approach is similar to the visual odometry approach except that is uses the surface normal instead of the 3D locations of the features. The steps for this algorithm are as follows:

- Extract features
- Fit planar surfaces to features using stereo depths
- Choose features on most planar surfaces
- Track these features using affine matching
- Compute an estimate of the delta pose from surface normals

Given that the surface normals are more stable than the feature locations, this algorithm will yield a more robust and accurate estimate of the pose change. Locating flat regions in the 3D map make good candidate targets for instrument placement and drilling operations.

Figure 5 shows a 3D textured mapped image and its densely calculated surface normals from stereo data. To reduce the amount of computation, we restrict the surface fitting to the extracted corner features of the images. This will retain the best planar fit features and eliminate points that will be hard to track. Alternatively, one can reverse the order of this operation—one can extract the best planar patches first, and then look for the most textured planes. However, the first approach is computationally more efficient.

Affine tracking with gain and offset

Similar to the visual odometry and the tracker algorithms, we use an affine tracker [8][15] to track features between two consecutive frames. Calculating affine distortions may reduce or eliminate the effects of geometric distortions, but changes in surface orientations also introduce photometric variations between the same regions in two images. These variations are often modeled in terms of gain and offset. For affine matchers—or their earlier predecessors such as KLT [8] that use intensity derivatives—offset values do not pose a problem. But the effects of the gain still remain. Figure 10 shows the resulting drift caused in tracking as a

function of variations in gain between a stereo pair from 0.5 to 1.5[3].

To overcome this problem, which is common in outdoor imagery, we incorporated estimating gain and offset along with the 6D affine parameters. The affine kernel can be used with different modes: x, y matcher; x, y, rotation matcher; 6D affine matcher; and 8D matcher with photometric balancing.

Pose Estimation from Surface Normals

Unlike the visual odometry algorithm described earlier which uses the 3D location of points to recover change in rover pose, this algorithm uses the surface normal information at the feature points. Horn [4][5] showed a closed form solution to the change in camera pose from surface normals.

Olson [13] showed that the largest contributor to errors in visual odometry and rover pose drift is errors in the rotation estimation. Motivated by this fact, we decided to examine the calculated surface normals as well as the position of the tracked points to get a better estimation of rover's pose. The rationale for this approach is as follows. We can detect outliers among 3D points in the visual odometry by looking at distances among 3D points which violate a rigid world assumption. Those with large relative displacements are outliers. But drifts of points collectively due to systematic errors may still occur.

On the other hand, surfaces and lines—being extended features—will not suffer from such limitations and will have smaller absolute errors than simple point tracking. Additionally, to calculate the relative orientation of two frames only requires two surface normals. Calculating translation however still requires three planar patches.

One can derive this formulation as follows. Assume we have two planes at time t that have normals u and v. At time t+1, these normals change orientation resulting in u' and v'. If the rotation between time steps is given **R,** we then have $u' = \mathbf{R}u$ and $v' = \mathbf{R}v$. But we can use the cross products $w = u \times v$ and $w' = u' \times v'$ to solve for the relative orientation matrix **R**. Furthermore we would like to calculate a rotation matrix **R** that maximizes the following function:

$$f(\mathbf{R}) = u'^{T}\,\mathbf{R}\,u + v'^{T}\,\mathbf{R}v + w'^{T}\,\mathbf{R}w$$

subject to $\mathbf{R}^{\mathrm{t}}\mathbf{R}$ being the identity matrix **I**. The superscript T denotes the matrix transpose.

One solution, often used in factorization based structure from motion, is to solve for the nine elements of the **R** matrix using least squares and then enforce the orthonormality of the matrix by using the QR factorization on the result to find the best orthonormal estimate for **R**. While extremely fast this approach is susceptible to degradation with noise.

A more accurate approach is to find the rotation matrix, **R** that maximizes the above expression and satisfies the constraint: $\mathbf{Q} = \mathbf{I} - \mathbf{R}^{\mathrm{T}}\,\mathbf{R} = \mathbf{0}$, where **I** is the identity matrix and **0** is the zero matrix. Using a Lagrange multiplier approach, we can maximize the expression:

$$G(\mathbf{R}) = \frac{d}{d\mathbf{R}}(f(\mathbf{R}) - 0.5\ \mathrm{Trace}(\mathbf{M}(\mathbf{R}^{\mathrm{T}}\mathbf{R} - \mathbf{I})) = 0$$

where the matrix **M**, similar to the matrix **Q**, which we like to minimize (hence the negative multiplier used to maximize expression for G) is a symmetric matrix of Lagrange multipliers and 0.5 is a constant used for convenience. The trace takes care of the element by element multiplication of the Lagrange multipliers and the elements of **Q**. Using the formulas $\mathrm{Trace}(\mathbf{A}\,\mathbf{B}) = \mathrm{Trace}\,(\mathbf{B}^{\mathrm{T}}\,\mathbf{A}^{\mathrm{T}})$ and $d/d\mathrm{A}(\mathrm{Trace}(\mathbf{AB})) = \mathbf{B}$, we further calculate G (**R**) to be:

$$u'u^{T} + v'v^{T} + w'w^{T} - \mathbf{M}\mathbf{R}^{T} = \mathbf{K} + \mathbf{M}\mathbf{R}^{T} = 0$$

which leads to:

$$\mathbf{M} = \mathbf{K}\mathbf{R}$$

or alternatively:

$$\mathbf{R} = \mathbf{K}^{-1}\mathbf{M}$$

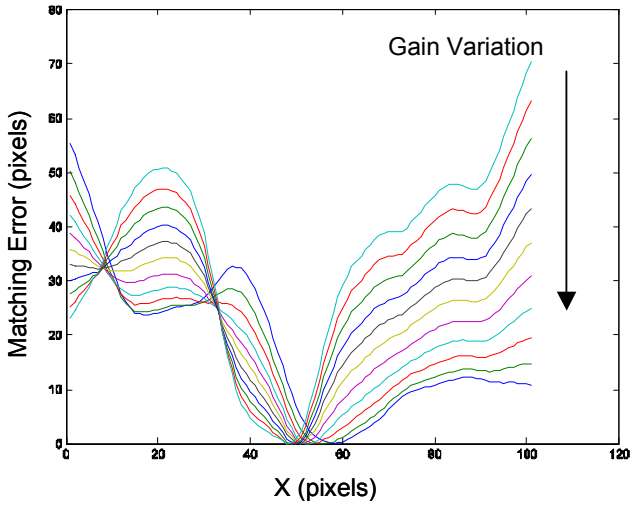

Figure 10: Variations in gain between image regions produce drift on the position of the minimum—*i.e.* the best match

[3] In reality gain also has a secondary effects involving the deformation of the correlation surface and its shifts upward. This vertical shift was removed since it has no effect in the translational drift.

With the **K** matrix being a known 3x3 matrix that can be summarized as:

$$\mathbf{K} = \sum_{\text{Surface Normal}} n_i \, n_i'^{T}$$

where n_i and n'_i are surface normals at t and $t+1$ respectively. Noting that **M** is symmetric and that $\mathbf{R}^T\,\mathbf{R}$ is identity, where:

$$\mathbf{R}^T\mathbf{R} = \mathbf{I} = \mathbf{M}\mathbf{K}^{-1}(\mathbf{K}^{-1})^T\mathbf{M}$$

which leads to

$$\mathbf{M}^2 = \mathbf{K}\,\mathbf{K}^T$$

that is **M** is the symmetric square root of the $\mathbf{K}\,\mathbf{K}^T$. If we write **K** in terms of its SVD decomposition $\mathbf{U}\,\Sigma\,\mathbf{V}^T$, with **D** the diagonal matrix of non-negative singular values, then

$$\mathbf{M} = \mathbf{U}\Sigma\,\mathbf{U}^T$$

Using the above **M** in $\mathbf{R} = \mathbf{K}^{-1}\,\mathbf{M}$ we get $\mathbf{R} = (\mathbf{U}\,\Sigma\,\mathbf{V}^T)^{-1}\,\mathbf{U}\,\Sigma\,\mathbf{U}^T$ which leads to the final solution:

$$\mathbf{R} = \mathbf{V}\,\mathbf{U}^T$$

One can easily verify that the constraint $\mathbf{R}^T\,\mathbf{R} = \mathbf{I}$ is satisfied.

What makes this approach interesting is its simplicity and speed. Given matching surface normals, or matching point clouds distances from their centroids, or the direction of matching lines in two frames of reference, once can easily construct the 3x3 matrix **K** and calculate the rotation matrix **R** from the two components of its SVD.

Initial testing of this algorithm on simulated data indicate that this formulation is robust to noise to the presence of noisy data constituting a planar patch. Further comparative analysis with traditional techniques are currently under way.

7. SUMMARY

We have presented a number of techniques that were studied to enhance the tracking of a designated target from a rover platform in Mars-like conditions. Our conclusion regarding the use of ICP was that it will not lead to enhancements in the accuracy of the tracked target because of the absence of a good initial estimate and the presence of noisy stereo data at 10 m range. Alternatively, the use of visual odometry to seed the designated target tracker yields good results. The 2D/3D target tracker used a normalized cross correlation between consecutive frames and an affine refinement of the new feature location relative to the original designated target at every step. The use of an adaptive view-based matching technique will increase robustness by recovering from failures in obtaining an estimate of the motion from the visual odometry. This occurs when the rover falls off a rock. Tracking with 4 mm navigation camera is robust but not accurate enough to lead to 1 cm error at the end of a 10 m traverse. Tracking the target with 60° FOV cameras and seeding this information to the mast to point the 17° FOV cameras for a high accuracy tracking produces the best overall results. So the 2D/3D tracker achieves desired accuracy by tracking with two cameras with different lenses. The use of surface normals to filter out features that feed the visual odometry did not produce any improved results. However, computing the change in rover pose from surface normals holds some promise. The reported results were from tests conducted on the Rocky 8 rover at JPL and the K9 rover at ARC.

8. ACKNOWLEDGMENTS

The work described in this paper was carried out under the competed task “Combining 2D/3D Visual Information for Target Tracking” within the Regional Mobility and Surface Access Mars Technology Program. This work was carried out at the Jet Propulsion Laboratory, California Institute of Technology, under contract to the National Aeronautics and Space Administration and at NASA Ames Research Center.

9. REFERENCES

[1] P.J. Besl, and H.D. McKay, “A method for registration of 3-D shapes.” *IEEE Transactions Patt. Analysis and Machine Intelligence* (PAMI); 14(2), pp 239-256 1992.

[2] M.C. Deans, C. Kunz, R. Sargent, L. Pedersen, “Terrain model registration for single cycle instrument placement,” *International Symposium on Artificial Intelligence,*
Robotics and Automation in Space, Nara, Japan, May, 2003

[3] M. A. Fischler and R. C. Bolles, ”Random sample consensus: a paradigm for model fitting with application to image analysis and automated cartography”, *Communication Association and Computing Machine*, 24(6), pp.381-395, 1981

[4] B.K.P. Horn, "Closed-form solution of absolute orientation using unit quaternions", *JOSA 4*(4), pp 629—642, April 1987.

[5] B.K.P. Horn, H. Hilden and S. Negahdaripour, “Closed-form solution of absolute orientation using orthonormal matrices", *JOSA 5*(7), pp 1127—1135, July 1988.

[6] T. Huntsberger, "Rover Autonomy for Long Range Navigation and Science Data Acquisition on Planetary

Surfaces," *IEEE International Conference on Robotics and Automation*, Washington, DC, May 2002.

[7] J. P. Lewis, "*Fast normalized cross-correlation,"* In Vision Interface*,* 1995

[8] Bruce D. Lucas, Takeo Kanade. "An Iterative Image Registration Technique with an Application to Stereo Vision", *Proceedings of the DARPA Image Understanding Workshop*, 1981.

[9] L. Matthies, "Dynamic Stereo Vision," *Ph.D. Dissertation*, CMU-CS-89-195, October 1989.

[10] Louis-Philippe Morency, Ali Rahimi, Trevor Darrell. "Adaptive View-based Appearance Model", *IEEE Conf. on Computer Vision and Pattern Recognition*, 2003.

[11] I. A. Nesnas, M. W. Maimone, H. Das, "Autonomous Vision-Based Manipulation from a Rover Platform," Proceedings of the 1999 *IEEE International Symposium on Computational Intelligence in Robotics and Automation*, pp. 351-356, November 1999, Monterey, California.

[12] I.A. Nesnas, A. Wright, M. Bajracharya, R. Simmons, T. Estlin, Won Soo Kim, "CLARAty: An Architecture for Reusable Robotic Software," *SPIE Aerosense Conference*, Orlando, Florida, April 2003.

[13] Clark F. Olson. "Adaptive-Scale Filtering and Feature Detection Using Range Data", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2000.

[14] Clark Olson, Larry Matthies, Marcel Schoppers, Mark Maimone, "Robust Stereo Ego-motion for Long Distance Navigation," *IEEE Conference on Computer Vision and Pattern Recognition*, June 13-15, 2000

[15] Jianbo Shi, Carlo Tomasi. "Good Features to Track", *IEEE Conf. on Computer Vision and Pattern Recognition*, 1994.

[16] Wettergreen, D., Thomas, H., and Bualat, M., "Initial results from vision-based control of the Ames Marsokhod rover," *IEEE Int'l Conf on Intelligent Robots and Systems,* pp. 1377-1382, France, 1997

[17] Wilcox, B. and Nguyen, T. "Sojourner on Mars and Lessons Learned for Future Planetary Rovers," *Society of Automotive Engineers publication 981695*, 1997.

[18] Yalin Xiong, Larry Matthies. "Error Analysis of a Real Time Stereo System", *IEEE Conf. on Computer Vision and Pattern Recognition*, 1997.

BIOGRAPHIES

***Issa A.D. Nesnas**, Ph.D. is the principal investigator for the visual tracking task and the principal investigator for the Architecture and Autonomy Research collaborative task. His research interests include sensor-based robot control and software and hardware architectures for robotic systems. Issa received a B.E. degree in Electrical Engineering from Manhattan College, NY, in 1991. He earned the M.S. and Ph.D. degrees in Mechanical Engineering from the University of Notre Dame, IN, in 1993 and 1995 respectively. Prior to joining JPL in 1997, he worked as a senior engineer at Adept Technology Inc. Issa is a member of Eta Kappa Nu and Tau Beta Pi National Honor Societies.*

***Max Bajracharya** received his Bachelors and Masters degrees in computer science and electrical engineering from MIT in 2001. He works on rover-based vision research and is currently a lead systems and software engineer for the CLARAty software project and the Rocky 8 research rover.*

***Esfandiar Bandari** recieved his BS and one Masters from UC Berkeley, and another Masters from Stanford University and his Ph.D. from University of British Columbia. He was a memeber of the research team at Vancouver General Hospital Medical Imaging Group, Mac Donald Dettwiler and Associates research team, and the General Reality Corp. Currently he is at NASA Ames Research Center in Mountain View, California.*

***Richard Madison** is a member of the Machine Vision group at JPL. His recent work includes 2D/3D tracking, modularization of in-house computer vision code, and vision-based object-pose-estimation for on-orbit rendezvous and capture. Previously, he has worked for the Air Force Research Laboratory and in industry, on various programs related to autonomous satellite servicing and semi-automated target recognition, respectively. He holds a BS in Engineering from Harvey Mudd College and MS and PhD in Electrical*

and Computer Engineering from Carnegie Mellon University.

***Clay Kunz** is the lead software engineer for the K9 rover at NASA Ames. He is also the head of the math and data structures subgroup of CLARAty. He's been an employee of QSS Group, and has had his hands inside K9, at Ames since 2001, before which he spent time making robot tour guides at a start-up company in Pittsburgh, PA. Clay holds BS and MS degrees from Stanford University, and lives in San Francisco.*

***Matthew Deans** earned a BS in Physics and a BS and MS in Electrical Engineering at Lehigh University, followed by a PhD. in Robotics fromCarnegie Mellon University in 2002. He is now working with the Autonomy and Robotics Area at NASA Ames Research Center under contract with QSS Group, Inc. His work has primarily focused on vision and localization for planetary rovers.*

***Maria Bualat** is the project manager and lead systems engineer for the K9 rover an NASA Ames Research Center. Maria received her BSEE from Stanford University in 1987, and her MSEE, emphasis Control Systems, from Santa Clara University in 1992. She has been a researcher on vision and navigation tasks for mobile robots since 1996. Prior to working on robots, Maria was a member of the Ames Photonics research group, developing optical matrix and fourier processors and fiber optic microphones*