

# VISION-BASED LOCALIZATION AND TERRAIN MODELING FOR PLANETARY ROVERS

*Timothy Barfoot, Space Missions, MDA, Canada*  
*Stephen Se, Space Missions, MDA, Canada*  
*Piotr Jasiobedzki, Space Missions, MDA, Canada*

## 1. INTRODUCTION

As discussed throughout this book, the next round of planetary missions will require increased autonomy to enable exploration rovers to travel great distances with limited aid from a human operator. For autonomous operations at this scale, localization and terrain modeling become key aspects of onboard rover functionality. Previous Mars rover missions have relied on odometric sensors such as wheel encoders and inertial measurement units/gyros for on-board motion estimation. While these offer a simple solution, they are prone to wheel-slip in loose soil and drift of biases, respectively. Alternatively, the use of visual landmarks observed by stereo cameras to localize a rover offers a more robust solution but at the cost of increased complexity. Additionally rovers will need to create photo-realistic three-dimensional models of visited sites for autonomous operations on-site and mission planning on Earth.

Our chapter begins with a formulation of the problem under investigation. We attempt to show the need for vision-based localization and modeling by examining the requirements of an upcoming Mars mission. We in turn formulate our own requirements that are used to guide the work to be described subsequently.

We proceed by presenting computational intelligence techniques developed at MDA which employ a stereo camera to observe Scale Invariant Feature Transform (SIFT) features both nearby and on the horizon to localize a rover. Field trial results are provided that show it is possible to reduce localization errors to a few percent of distance traveled, a major improvement over

conventional odometric sensors. Comparisons are made to similar techniques in the literature. Some discussion is given to the computational burden of including such a technique on future planetary missions. As part of the solution, we have implemented aspects of the vision processing algorithms on Field Programmable Gate Array (FPGA) processors, which serve to speed up computations for online localization.

Next we present a technique developed at MDA to create photo-realistic three-dimensional models from stereo image sequences. The technique is voxel-based with color texture mapping. The relative localization of image over time is provided by the SIFT-based localization technique described above and thus model creation can be easily integrated into our framework. The resulting 3D models require very little mass storage and transmission bandwidth compared to the original images, and thus are good candidates for sending back to Earth-based operators.

The chapter concludes with recommendations for further development and a summary of open problems in this area.

## **2. PROBLEM FORMULATION**

In this section we introduce the problem under investigation and attempt to show its importance to upcoming planetary missions. In particular, we examine some requirements of the European Space Agency's 2011 ExoMars mission and show how vision-based localization and modeling may play a key role in achieving the autonomy necessary to conduct the mission.

### **2.1 Example Mission Requirements**

The ExoMars Rover is a key element of the ExoMars mission, the first flagship mission of the Aurora Programme initiated by the European Space Agency (ESA). The aim of this programme is to characterize in detail the Mars biological environment in preparation for future missions, including human exploration. Carrying a large suite of exobiology instruments, the ExoMars Rover will be capable of operating autonomously, traveling several kilometers over rocky Martian terrain, and drilling to collect samples for analysis by the instruments. Planned for launch in 2011, the main purpose of the ExoMars mission is to search for signs of past and present life on Mars. In a Phase A study performed for ESA, MDA led an international industrial team to develop an optimized conceptual design of the Rover, incorporating specialized electrical power generation, thermal control, navigation, telecommunications and vehicle control subsystems (see Figure 1).

Table 1 lists some of the requirements for the ExoMars rover taken directly from ESA's System Requirements Document [1]. Based on the requirements of this mission, it can be worked out that the nominal mission duration will be 60 sols, with 6 sols per experiment cycle. Accounting for the time to carry out the scientific data collection, it can be determined that 2 sols will be available to traverse 2 km between sampling locations. If roughly half the 2 km distance is traversed each day, this implies a single mid-course correction can be provided by ground operators, but otherwise the rover must function completely autonomously. Thus, the onboard guidance, navigation, and control algorithms must be able to move the rover to a new position 1 km away. The ability to localize using on-board functionality is a critical skill.

This in itself does not imply that a vision-based localization and modeling capability is necessary. In the next section we briefly look at other techniques that could be used to position a rover on Mars.

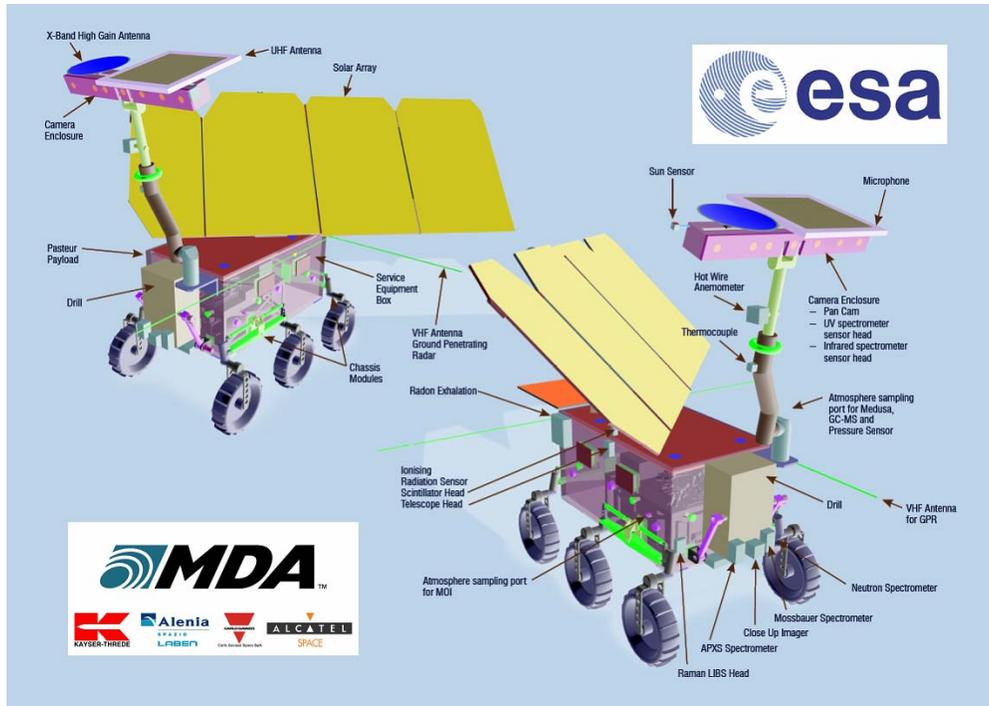


Figure 1: The European Space Agency's 2011 ExoMars mission provides an example of a next-generation Mars rover that will require advanced computational intelligence techniques to achieve its mission goals. A team led by MDA developed the proposed design for the ExoMars rover shown here.

The Rover shall be able to perform at least 10 Experiment Cycles on the Martian Surface as a Nominal Rover Mission.
The Rover should be able to travel 2 km between consecutive sampling locations.
The Rover design shall be compatible with only one command cycle from Earth per sol.
The Rover shall be able to navigate autonomously through the rough terrain defined hereafter.
The PanCam shall allow the identification of the rover's position.
The PanCam shall allow the determination of the rover's orientation and tilt attitude.
The PanCam shall support the planning and determination of rover traverse operations.
The PanCam shall support the construction of local digital terrain models, generated with PanCam data.
The Rover shall not require any ground feedback loops shorter than 24 hours duration.
During nominal operations the Rover shall be able to continue to operate without ground contact for a period of 48 hours without interrupting mission product generation.
Rover localization shall be performed, providing the 3 axes attitude angles with 1-degree accuracy.

Table 1: A sampling of requirements for the ExoMars Rover that drive the need for autonomy and vision-based computational intelligence [1].

## 2.2 Existing Mars Localization Techniques

At this stage we must make the distinction between absolute and relative localization. Known absolute localization techniques such as radiolocation and horizon feature matching to elevation

data provide updates too infrequently to be used throughout a sol [2]. These techniques are more appropriate to making corrections at the end of a sol or every few sols. On Earth we have access to the global positioning system but for planetary applications this is not yet available.

To localize throughout a sol we typically require a relative localization system that tries to estimate the pose of a rover relative to a reference frame attached to the initial pose of the robot. No attempt is made to find the correspondence between the initial reference frame and a global reference frame. The frequency of updates from a relative localization system is typically much higher than an absolute localization system. The rest of this section discusses common relative localization techniques.

Wheel odometry estimates the velocities of each wheel and since they are part of the motion control components, utilizing these sensor data is relatively simple and inexpensive. The velocities are integrated over time to produce a position estimate. However, wheel odometry is extremely prone to slip on natural terrains and this in turn affects the attitude estimate. Many studies have proven that localization relying on odometry alone can produce 20%-25% error of distance travel in position estimate [3]. The 2003 Mars Exploration Rovers experienced considerable slip on occasion, corrupting odometry measurements.

Inertial measurement units (IMU) can provide translation and attitude of a rover by using of 3-axis accelerometers and 3-axis gyroscope rate sensors. Both accelerometers and gyros can however be influenced by bias errors which can lead to unbounded growth in error over time. Bias fluctuations over an entire sol preclude using an IMU alone.

Improved orientation estimates can be obtained by employing a Sun sensor, which compares a detected sun vector with internal knowledge of Sun's expected location based on ephemeris data. The inclusion of a Sun sensor was one of the main recommendations after the 1997 Mars Pathfinder mission [4]. On hard terrain, a Sun sensor in combination with odometry can provide a cheap relative positioning device [5]. For baseline operations, 2003 Mars Exploration Rovers employed Sun sensors in combination with other sensors, but estimates of translation relied heavily on odometry for which slip was a problem on loose terrain.

Stereo cameras have been present on planetary rovers for other purposes, namely obstacle detection and avoidance, and modeling. However, only the recent Mars Exploration Rovers have used visual odometry as a technology demonstration (discussed below). The slow acceptance of vision-based localization may be due to limited computational resources and power.

### **2.3 Problem Statement**

Based on knowledge of past rover missions and the anticipated requirements for future rovers to travel longer distances per sol and to generally perform more autonomously, we make the assertion that an improved relative localization system will be needed. There are several possible avenues that might be pursued to provide such a system. However, given that baseline rover operations already rely on stereo cameras for obstacle avoidance and modeling, it is logical to attempt to use these same sensors to help improve localization. This brings us to the top-level problem statement:

**Develop a vision-based localization system to allow a planetary rover to position itself with errors limited to a few percent of distance traveled over a several kilometer traverse across unknown terrain.**

A consequence of solving the above problem is that it facilitates the solution to another problem: creating a high-resolution three-dimensional terrain model of the environment for visualization and planning. Our major focus in this chapter will be on addressing the localization problem but as we will see, piggy-backing a vision-based terrain modeling technique is a natural extension and thus will also be examined.

### 3. SYSTEM OVERVIEW

In this section we present our approach to vision-based localization and terrain modeling. A graphical depiction of the data flow in our system can be seen in Figure 2. We will describe the blocks in this diagram in detail below, but some general comments can be made. As the title of this chapter implies, stereo imagery is used for two purposes: localization and terrain modeling. As we can see there are some steps that are common to these two goals, namely image capture and undistortion/rectification. Moreover, we can see from Figure 2 that terrain modeling relies on the output of visual motion estimation. This is because we seek to create terrain models from a moving platform and so data from images taken in different locations must be merged. Finally, although motion planning and obstacle avoidance are beyond the scope of our discussion, we point out that the 3D terrain map can be used for two purposes: situational awareness and autonomous motion planning. To accomplish motion planning the terrain map would first be converted to a cost map. Alternative terrain estimation techniques are discussed in [6].

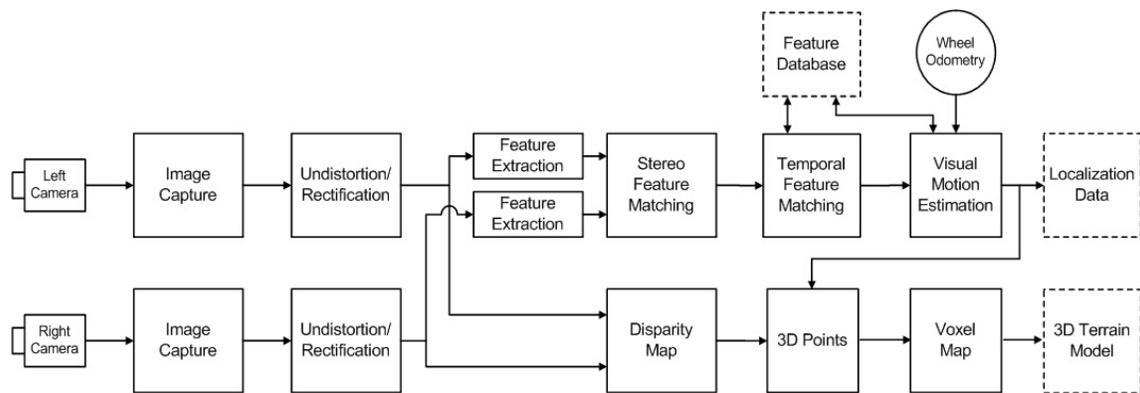


Figure 2: Dataflow diagram for vision system.

The remainder of this section will be devoted to explaining some of the key blocks of Figure 2 in more detail. In the interests of brevity, we will not delve into the details of Image Capture nor Undistortion/Rectification. We state simply that the function of the Undistortion/Rectification block is to transform the raw images into a rectified form that simplifies stereo processing. At the highest level, the intent is to make the images appear as though they had been produced by a stereo camera consisting of two pinhole cameras that are perfectly aligned with one another. This requires that the stereo camera undergo a calibration procedure in advance of use to determine the appropriate transformation.

To illustrate some of the other blocks of Figure 2 we will adopt a running example. As our intended application is vision for a planetary rover, we will use two sets of images obtained by the Mars Exploration Rover, Spirit. These images, shown in Figure 3, were taken by Spirit's Front HazCam on Sol 15 of the primary mission as it approached a rock feature called "Adirondack". These images are 1024x1024 pixels and the relevant stereo camera parameters can be found in the literature. Figure 3 also shows what the images look like after Undistortion/Rectification. Both the raw and undistorted/rectified images were obtained from the MER Analyst's Notebook web repository [7].

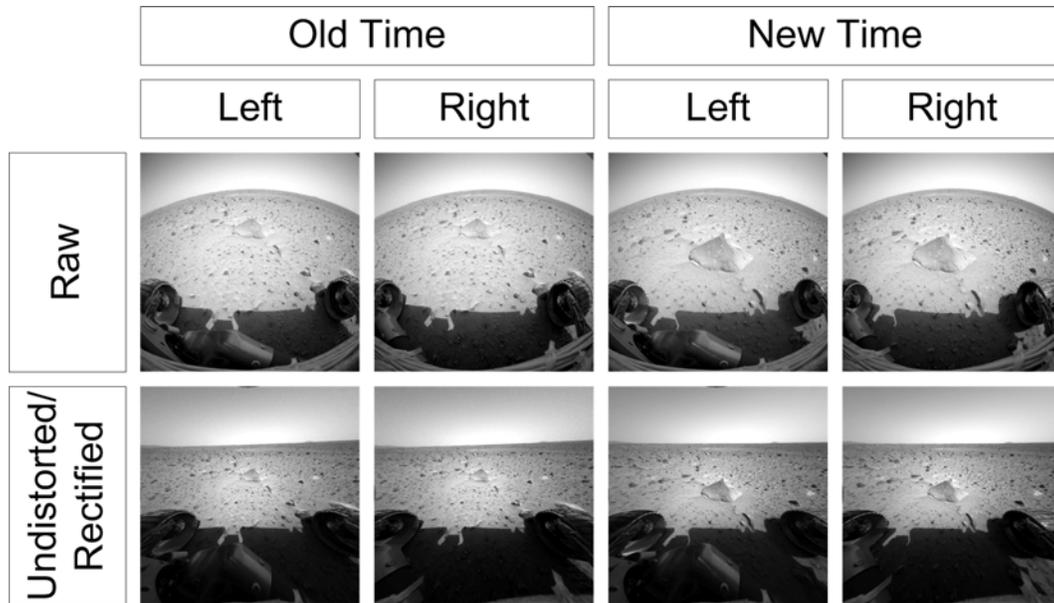


Figure 3: Two stereo pairs from the Mars Exploration Rover, Spirit. These images were taken on Sol 15 as Spirit approached the rock feature dubbed "Adirondack". Courtesy NASA/JPL-Caltech.

### 3.1 Feature Extraction

In our implementation of vision-based localization, we attempt to automatically identify and track a large number of visual landmarks, or features as the rover moves. We have chosen to use a high level set of natural visual features called Scale Invariant Feature Transform (SIFT) as the visual landmarks to compute the camera motion. SIFT was developed by Lowe [8][9] for image feature generation in object recognition applications. The features are invariant to image translation, scaling, rotation, and partially invariant to illumination changes and affine or 3D projection. These characteristics make them suitable as landmarks for robust matching when the cameras are moving around in an environment. Such natural landmarks are observed from different angles, distances or under different illumination.

Previous approaches to feature detection, such as the widely used Harris corner detector [10], are sensitive to the scale of an image and therefore are less suitable for building feature databases that can be matched from a range of camera positions. A comparison between Harris corners and SIFT features is shown in Table 2. The SIFT features are detected by identifying repeatable points in a pyramid of scaled images. Feature locations are identified by detecting maxima and minima in the Difference-Of-Gaussian pyramid. A subpixel location, scale and orientation are associated with each SIFT feature. In order to achieve high specificity, a local feature vector [8] is formed by measuring the local image gradients at a number of orientations in coordinates relative to the location, scale and orientation of the feature. The local and multi-scale nature of the features makes them insensitive to noise, clutter and occlusion, while the detailed local image properties represented by the features make them highly selective for matching to large databases.

The function of the Feature Extraction block is to extract a large set of SIFT feature from a single image. Figure 4 shows the over 2000 SIFT features that were identified in Left and Right Old Time images in our example. Each feature is marked with a white box. The size of the box represents the scale of the feature while the rotation of the box represents the orientation of the feature. It is worth noting that features were found at many scales and orientations both near the rover and out to the horizon.

Although SIFT features are reasonably unique in their description as compared to Harris corners, there is an added computational burden associated with their use. For this reason, we have implemented the Feature Extraction block on a Field Programmable Gate Array (FPGA). Because this is an implementation detail it will be described below in the section on implementation.

	<b>Harris Corners</b>	<b>SIFT Features</b>
Algorithm complexity	Easy to detect	Complex detection algorithm
Localization accuracy	Sub-pixel	Sub-pixel
Scales	Single or multiple scales	Multi-scale representation
Description	Image windows	Specific local image feature vector
Correspondence	Hard, many mismatches	Easy, few mismatches

Table 2: Comparison between Harris corners and SIFT features.

### 3.2 Stereo Feature Matching

With known stereo camera geometry, the SIFT features in the left and right images are matched using the following criteria: epipolar constraint, disparity constraint, orientation constraint, scale constraint, local feature vector constraint and unique match constraint [11]. All of these constraints are essentially inequality-type constraints with tunable thresholds. By varying the thresholds we may trade off the number of features against the quality of matched features.

Because we have exploited the stereo camera geometry, the quality of the matched features is typically very high coming out of the Stereo Feature Matching block. Figure 4 (right) shows the over 1000 stereo-matched SIFT features that were found for the Old Time stereo pair in our example. The image shown is the Right image for this pair and each match is marked with a white arrow. The tail of the arrow is at the location of the feature in the Right image and the head of the arrow is at the location of the feature in the Left image. Thus, the length of the arrow represents the disparity between the Left and Right images. We can see that the lines are all horizontal as expected due to the epipolar constraint and disparity is larger close up and smaller far away. We can also note that matches were only found in the center region of the image. One cause of this can be an imperfect calibration/undistortion/rectification process in the image periphery whereby the epipolar and disparity constraints can help avoid making unwanted matches away from the center region.

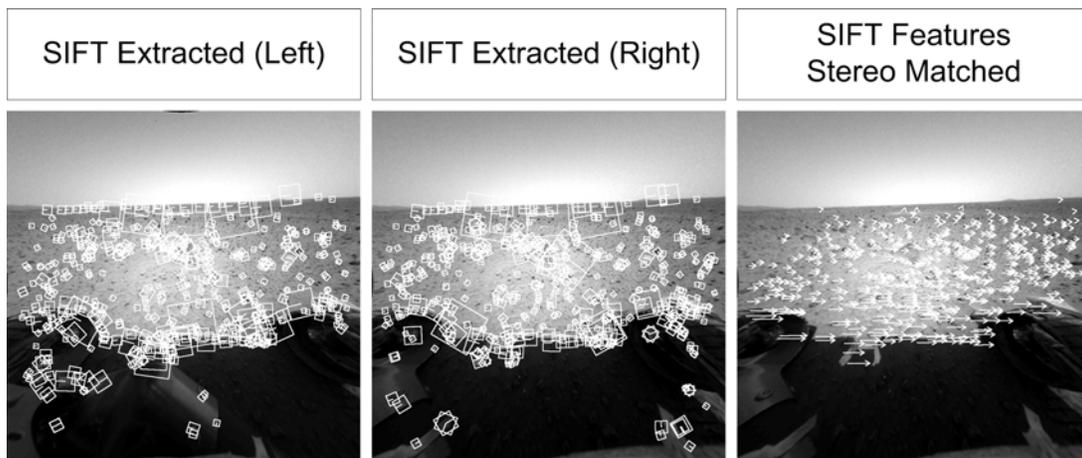


Figure 4: Stereo matching of SIFT features between the Left and Right images at the Old Time in Figure 3.

### 3.3 Temporal Feature Matching

Temporal feature matching is typically done in one of two ways: single frame or multi-frame. In single frame matching, a stereo pair is compared only with the previous frame. In multi-frame matching a database is built up and the current frame is compared to the database. Other authors [12] have reported a 28% reduction in rover navigation error when multi-frame matching is used, rather than considering each pair of frames separately. Our preferred approach is to maintain a database, but for the planetary application a trade study should be performed to select the most appropriate technique. This is because there is a cost in terms of the additional memory and computational cycles needed to maintain and search the database.

The function of the Temporal Feature Matching block is to take each of the stereo-matched features from the current frame and find the best match in our growing database of features. If a feature cannot be found in the database, we add it and assign it an id number. To maintain fast access in our implementation, a kd-tree is built online and matching of observed features to the database (i.e., data association) is carried out by a best-bin-first search [13]. We have experimented with database sizes up to 200,000 features.

The actual temporal matching criteria are: column constraint, row constraint, orientation constraint, scale constraint, local feature vector constraint, unique match constraint [11], and temporal constraint. The column and row constraints can be used if something about the camera motion is known (e.g., forward movement). The temporal constraint can be used to limit how far back in time a match is allowed. The unique match constraint, however, is typically the most important one and can actually be used in isolation if necessary. Again, applying stricter constraints can allow one to trade number against quality of matches.

Figure 5 (bottom) shows some temporally-matched features for our running example. The image shown is the Right/New Time image and temporally-matched and the line connecting the previous position (tail) to the current position (head) is analogous to optical flow. Here roughly 100 temporal matches were found and we can see the movement of these features is qualitatively correct as the rover moved forward from the Old Time to the New Time.

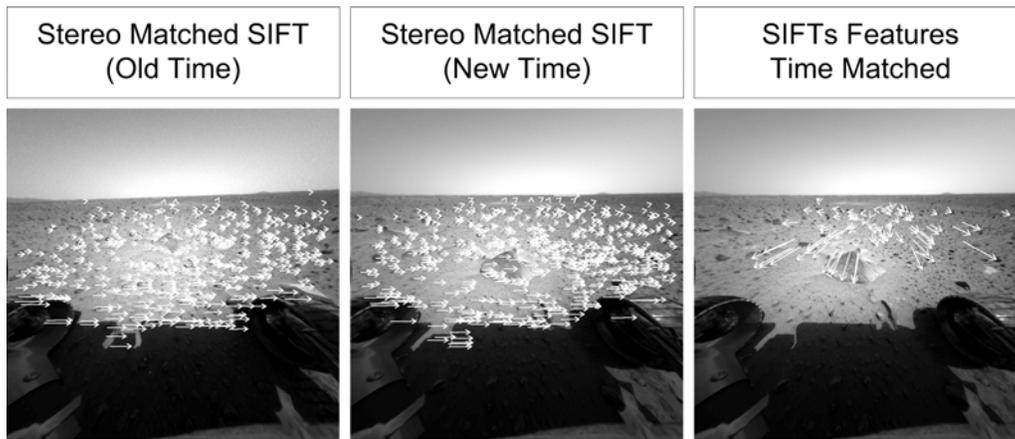


Figure 5: Temporal matching of SIFT features between New and Old stereo-matched features.

### 3.4 Visual Motion Estimation

Once we have completed all of the feature extraction and feature matching steps, we seek to estimate the full three-dimensional motion of the camera from the resulting features. To do this we employ a Simultaneous Localization and Mapping (SLAM) approach that has the following essential steps:

1. Predict the camera motion using odometric sensors.
2. Correct this camera motion using the observations of SIFT features that have been temporally-matched to the database. This is done using a weighted least squares technique that accounts for the feature uncertainty.
3. Update the features in the database (our map) using the final camera motion.

In more detail, our estimation algorithm is derived from the FastSLAM 2.0 algorithm [14][15]. Some modifications were necessary to make the algorithm compatible with our scenario [16]. The biggest change to the original algorithm is that we observe a large number,  $K$ , of SIFT landmarks simultaneously (e.g.,  $K = 500$ ). Unlike other feature-based SLAM work, we typically build databases with tens of thousands SIFT landmarks and therefore, standard Extended Kalman Filter approach does not work as it cannot handle the monolithic covariance. We also needed to incorporate outlier detection as some of the visual landmarks are inevitably mismatched.

We seek to simultaneously estimate the trajectory of a vehicle as well as the states of  $L$  landmarks. Mathematically this is expressed as the joint probability density for the vehicle trajectory and landmarks positions, given all the observations:

$$p(\mathbf{s}^t, \mathbf{x}_1, \dots, \mathbf{x}_L | \mathbf{z}^t, \mathbf{u}^t, \alpha^t) = p(\mathbf{s}^t | \mathbf{z}^t, \mathbf{u}^t, \alpha^t) \prod_{l=1}^L p(\mathbf{x}_l | \mathbf{s}^t, \mathbf{z}^t, \mathbf{u}^t, \alpha^t) \quad (1)$$

which we see can be factored into  $L$  landmark state-estimators and one vehicle trajectory estimator. The vehicle states, up to time  $t$  (a.k.a., its trajectory up to time  $t$ ), is denoted  $\mathbf{S}^t$ . The  $l^{th}$  landmark state is denoted  $\mathbf{x}_l$  (which is assumed to be stationary). The sensor observations, up to time  $t$ , are denoted  $\mathbf{Z}_t$ . The control inputs (or odometry measurements), up to time  $t$ , are denoted  $\mathbf{u}^t$ . The data associations, which assign particular observations to particular landmarks, up to time  $t$ , are denoted  $\alpha^t$ .

As is described in [14][15], a Rao-Blackwellized particle filter will be used to update the posterior as new observations are gathered. This type of particle filter uses samples to represent uncertainty in the vehicle trajectory. Within each particle (a.k.a., sample), an independent Kalman filter [17] is implemented for each landmark in the map. Thus for each landmark (in each particle) we are estimating a mean and covariance:

$$p(\mathbf{x}_l | \mathbf{s}^{(m),t}, \mathbf{z}^t, \mathbf{u}^t, \alpha^t) \sim N(\bar{\mathbf{x}}_{l,t}^{(m)}, \mathbf{C}_{l,t}^{(m)}) \quad (2)$$

where  $(m)$  is the particle index. This has the advantage of not requiring a monolithic filter to represent the joint density for the vehicle and all the landmarks. In our realtime implementation to date we have only been able to use a single particle to represent the vehicle trajectory. However, having formulated the problem in this way allows more particles to be added later if computational resources permit.

On the surface, it may seem that SIFT features implicitly handle the data association problem, allowing for a simpler estimation algorithm. This is true to a certain extent. However, the data association problem does not simply go away. Mismatches do occur and outliers must be detected to make Step 2 above work well. Fortunately, the percentage of outliers tends to be fairly small such that throwing away a few observations is tolerable. Our approach to handling outliers has a few steps:

- *Avoidance*: We sort the incoming observations based on the unique SIFT match criterion which tells us how good the SIFT match was and thus how good the data association was. We then begin processing the observations from best to worst, stopping after a predefined

number of non-outliers (e.g., 50). This is also a convenient way to limit the update time in feature-intensive scenes.

- *Probability*: For each SIFT observation, the probability that the estimated camera motion and the old feature position generated this observation is computed. We check that this probability is not too small. If it is, the observation is labeled an outlier.
- *Pose Change*: As a final resort, we check how much the estimated camera motion changes as a result of incorporating each SIFT observation. If it moves too far, the observation is labeled an outlier.

If an outlier is detected we undo the incorporation of that observation and proceed to the next one in the sorted list. Other outlier rejection techniques such as RANSAC [18] can be used instead and further evaluation is needed to compare the performance.

The output of the Visual Motion Estimation block is the full six degree-of-freedom pose of the robot. In our formulation, we are actually estimating the motion of a coordinate frame attached to the right camera of the stereo pair, rather than one attached to the robot base. This is purely a matter of efficiency; it is more efficient to transform a single odometry measurement to the camera frame than  $K$  visual landmark observations to the robot frame. It is a simple matter to transform the final camera pose back to the robot frame to provide an estimate of the robot motion. We represent the six degree-of-freedom pose of the camera using three numbers for position as well as four Euler parameters (a.k.a., quaternions) representing the rotation of the camera. It is important to note that there is a constraint on the Euler parameters such that only three of these are independent. One must be careful here because the constraint implies the space of robot poses is not a vector space. To handle this we must think of the linearization steps in the Kalman filters mentioned above as compounding a ‘small’ rotation vector with a ‘large’ mean rotation [19].

Note that if odometry measurements are used as the ‘control inputs’, one must carefully transform the associated covariance matrix from the robot frame to the camera frame, particularly if the camera is on a rotating pan-tilt unit, attached to the robot base. Table 3 quotes the motion estimation results for the running example.

<b>Longitudinal Movement</b>	<b>Lateral Movement</b>	<b>Rotation</b>
81 cm (forward)	3 cm (left)	0.5 degrees (clockwise)

*Table 3: Motion estimated purely from images for the running example. Three-dimensional results were projected to two dimensions.*

### 3.5 Disparity Map and 3D Points

To compute disparity maps offline, we use either Point Grey Research’s optimized Triclops library based on Sum of Absolute Differences (SAD) algorithm or MDA normalised correlation-based dense stereo algorithm [20]. Figure 6 shows the disparity map for the New Time image pair in our running example.

To compute disparity maps online for realtime applications, we use the 3DAware PCI card from Tyzx for dense stereo computation. It consists of a DeepSea2 chip, which is an optimized hardware implementation of the Census stereo algorithm [21]. As with other stereo algorithms, texture is required for stereo matching, and hence there is no match for uniform regions. The Tyzx system can compute dense stereo at 30Hz but is limited to 512x512 resolution.

Whether online or offline, a simple pinhole stereo camera model is used to reconstruct the dense 3D points from the disparity map. As the rover moves around, dense 3D points are obtained relative to the camera position at each frame. All data sets must be transformed to one reference coordinate system before they can be combined together. As mentioned in the section

on visual motion estimation, we have chosen to use the initial camera pose as the reference and all 3D data sets are transformed to this coordinate system using the camera pose estimated for each data set.

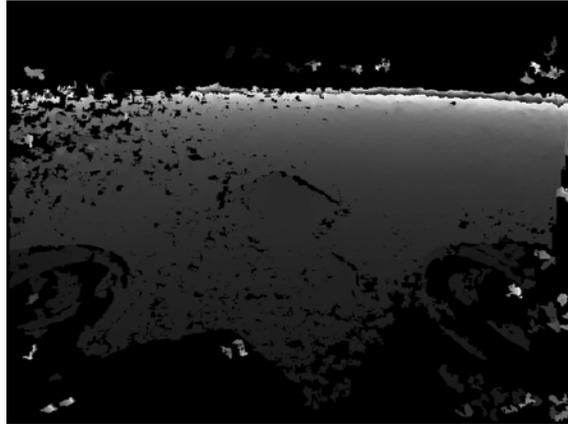


Figure 6: Disparity map for New Time image pair.

### 3.6 Voxel Map and 3D Terrain Map

Using all 3D points obtained from the stereo processing is not efficient as there are a lot of redundant measurements, and the data may contain noise and missing regions (due to incorrect matches or lack of texture). Representing 3D data as a triangular mesh reduces the amount of data when multiple sets of 3D points are combined and thus also reduces the amount of bandwidth needed to send the resulting models offboard (e.g., to Earth). Furthermore, creating surface meshes fills up small holes and eliminates outliers, resulting in smoother and more realistic reconstructions.

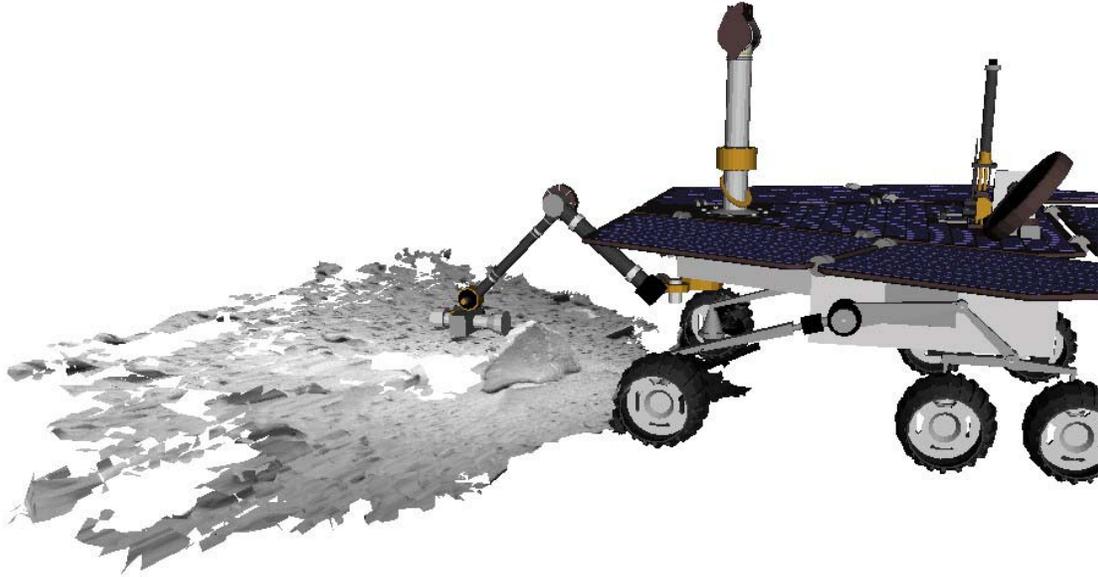
To generate triangular meshes as 3D models, we employ a voxel-based method [22], which accumulates 3D points with their associated normals. It creates a mesh using all the 3D points, fills up holes and works well for data with significant overlap. The 3D data is accumulated into voxels at each frame. Outliers are filtered out using their local orientation and by selecting the threshold of range measurements required per voxel for a valid mesh vertex. It takes a few seconds to construct the triangular mesh at the end, which is dependent on the data size and the voxel resolution.

Photo-realistic appearance of the reconstructed scene is created by mapping camera images as texture. Such surfaces are more visually appealing and easier to interpret as they provide additional surface details. Colour images from the stereo camera are used for texture mapping.

As each triangle may be observed in multiple images, the algorithm selects the best texture image for each triangle. A texture image is considered to be better if it is captured when the camera is facing the triangle directly. If the camera is looking at the triangle at an angle, then its quality is lower due to the lower and non-uniform resolution caused by perspective distortion. To find the best texture, the algorithm analyses all the images and selects the one that gives the largest area upon 2D projection according to the camera pose.

Moreover, we need to take into account any occlusion. For example, if there is an object in front of the triangle, then the image captured when the camera is facing the triangle directly should not be used, as the texture will be for the object in front. The algorithm then selects another texture image that gives the second largest area upon projection.

Figure 7 shows a textured terrain map for the New Time image pair in our running example. The “Adirondack” rock feature is clearly visible in three-dimensions when viewed from this perspective. A three-dimensional model of the Spirit rover was inserted for visualization.



*Figure 7: Terrain map with 3D rover model inserted for New Time image pair.*

## **4. IMPLEMENTATION**

### **4.1 Rover Testbeds**

We have used two different rover testbeds to date for hardware testing. Initial testing of our methodology was conducted using the rovers shown in Figure 8 (left). This rover consists of a 4-wheel chassis developed by the University of Toronto Institute for Aerospace Studies (UTIAS). The batteries of the rover are inside the tires to keep the center of gravity low. We have developed other hardware and software components to provide the capability of monitoring and controlling the rover in its environment remotely. A Bumblebee stereo camera from Point Grey Research (PGR) has been integrated into the rover system, with image capture up to 7 pairs per second, 8-bit 640x480 greyscale images, 70 degrees horizontal and 40 degrees vertical field of view, and Firewire interface. To increase the effective field of view of the camera, it is installed on top of a pan-tilt unit, for capturing multiple images. The limited computational power of this platform required that all images be uploaded to a ground station for processing.

Our second testbed is depicted in Figure 8 (right). The chassis of this rover is an iRobot ATRVJr with a custom vision system. The stereo camera was constructed using a pair of Sony DFW-X700 cameras, mounted on a rigid aluminum bar, affixed to a pan-tilt unit. The images we use are 8-bit 1024x768 resolution. The field of view of the cameras is approximately 45 degrees horizontal and 35 degrees vertical. There are currently two computers on board, a dual Pentium III 1 GHz with 1 Gb of RAM (inside the red box) and a dedicated vision computer consisting of a Pentium M 1.8 GHz with 1 Gb of RAM. The vision computer also houses our hardware accelerated vision processing boards, a Tyzx DeepSea2 for dense stereo calculations and an AlphaData ADM-XRC board with a Virtex II Xilinx FPGA running our implementation of SIFT feature extraction. There are various other sensors onboard as well: sonar rangefinders, SICK laser rangefinder, DGPS, compass, inertial measurement unit, inclinometer.



Figure 8: (left) Custom rover with Bumblebee stereo camera. (right) ATRV Jr rover with custom stereo camera.

## 4.2 FPGA Implementation

The high computational requirements of vision algorithms often limit the amount of distance and science that can be safely achieved by rovers equipped with radiation hardened processors. In order to speed up performance, we used dedicated hardware such as Field Programmable Gate Array (FPGA) for some intensive image processing to offload the processor.

For this work, we have implemented SIFT extraction on a Virtex II Xilinx FPGA as it is computationally intensive. The fixed point hardware implementation of SIFT was developed based on the floating point software version. To implement the complex SIFT algorithm directly using Very High-Level Design Language (VHDL) would have been a lengthy and time consuming task. A high level environment was needed.

System Generator is a software tool for modeling and designing FPGA-based signal processing systems in the Matlab-Simulink environment. Simulink provides a graphical environment for creating and modeling dynamical systems. System Generator consists of a Simulink library called a Xilinx Blockset, and software to translate a Simulink model into a faithful hardware realization of the model.

Even though the majority of the design was created with System Generator, there was coding in VHDL for low level processes that were not efficient to do with the Xilinx Block sets (such as DMA transfers, memory access routines and wrapper files). The System Generator design, low level VHDL coding and wrapper files were all brought into the Xilinx Integrated Synthesis Environment (ISE) software tool. The final bit file was generated within the ISE environment which then could be uploaded to the FPGA for execution.

Since the software uses floating point operations, testing was required to convert the software implementation to work with fixed point operations. Furthermore, many of the routines in the software version needed to be modified to make the hardware implementation efficient.

To extract SIFT features from a 640x480 image, it takes 600 ms for a Pentium III 700MHz processor, while the FPGA can do so within 60 ms and leave the processor available for other tasks.

## 5. EXPERIMENTAL RESULTS

### 5.1 Vision-Based Localization

Initially we tested our algorithm offline using rover simulations and image sequences from a satellite capture mockup scenario wherein a stereo camera approaches a free-flying satellite. Reasonable motion estimates were obtained for numbers of particles between 1 and 300. However, computation times for large numbers of particles were too high to provide an online estimation technique. Thus, for the moment, our visual motion estimation, implemented on the rover depicted in Figure 8 (right) uses a single particle ( $M=1$ ). This has the effect of not keeping the error correlations between landmark positions and the robot pose. As data association is carried out based on the SIFT features, the estimation is not affected adversely by the lack of particles. Our experiments revealed promising preliminary results, likely because our data association is reasonably good. Furthermore, increasing the number of particles still cannot guarantee consistent estimation due to the highly non-linear observation model.

We have field-tested the above algorithm running online both indoors and outdoors. Outdoors we conducted tests with the rover driving on two types of terrain: sand and gravel. The presence of loose terrain was a good test as it often causes odometry to become erroneous.

#### *Indoors:*

Initial testing was conducted indoors with varying success. Figure 9 (left) shows the results of an average traverse in a lab environment. Both the robot path and the resulting map are shown (projected to ground plane). In Figure 9 (right) we show an occupancy grid created using a laser rangefinder for comparison. In reality the robot's path was a closed loop but we see here that there is a large final error. Not surprisingly, there are also errors in the map (even compared to the laser map which itself is not perfect).

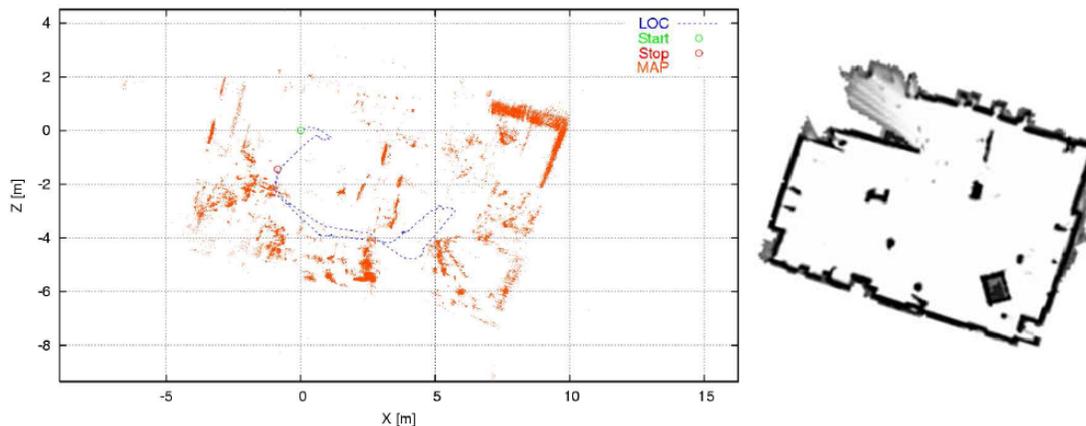


Figure 9: (left) Estimated rover localization and SIFT map (all projected to ground plane). (right) Occupancy grid created using laser-based technique for comparison.

Figure 10 shows how the number of visual observations changes throughout the traverse shown in Figure 9. From this plot we can begin to see the difficulty of using this technique in a cluttered, indoor environment. Often the camera got too close to objects to get good stereo matches (due to limits on maximum disparity) and the number of visual observations plummeted to zero. Similarly, if the camera was pointed at a blank wall, the number of visual observations would drop. With no visual observations, our system essentially defaults to simple odometry.

At the other end of the spectrum, if there are too many features, our system can also have trouble. Although we limit the number of observations of previously seen landmarks used in the

estimation (typically to 50), we still incorporate all new landmarks into the database. If there are a lot of new landmarks seen, then we are sometimes forced to drop camera frames in order to not let the localization fall too far behind.

We found that most realistic traverses indoors were confronted with periods of few or no features. This was particularly true for hallway traverses with large sections of blank walls.

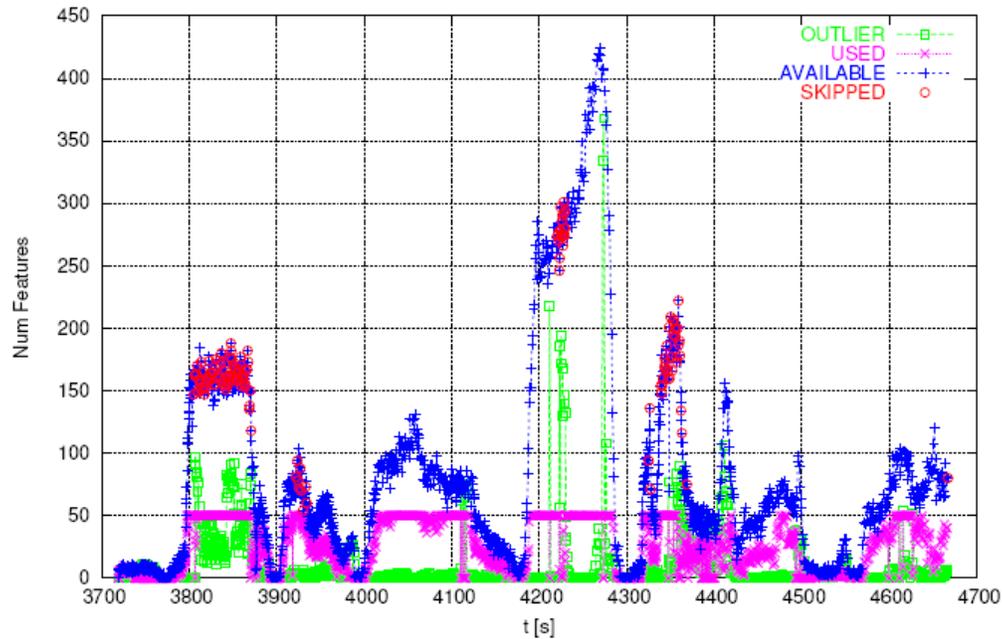


Figure 10: Statistics on number of visual features observed (blue), used in estimation (purple), outliers (green) during traverse shown in Figure 9. Red is used to indicate frames that had to be dropped to due to processing limitations.

#### Outdoors, Sand:

A set of field trials was conducted on a sandy surface (10 traverses total). Figure 11 shows the results of an approximately 40 m traverse at maximum speed of 5 cm/s on loose sand. During this run, the motion planning software chose to go left around an obstacle (8 m into the traverse). While executing this turn, a considerable amount of slip occurred, causing odometry to be quite erroneous in the orientation estimate. Our visual technique did a much better job of estimating the robot path, as can be seen by comparison to GPS. Using a tape measure, the final position of the rover was 37.8 m from the start. Visual motion estimation found 39.4 m and GPS found 38.8 m. The tape measure was taken as ground truth, indicating the visual motion estimation over-predicted the position by 4% of distance travelled. However, it should be noted that most of this positioning error was in the longitudinal direction (along the line joining the start and final positions). Visually, the lateral error was extremely small, indicating that orientation was estimated very well throughout the traverse.

Similar results were found for all the sandy traverses. The repeatability of the system was found to be quite high across trials. Qualitatively, this was demonstrated by the tracks that the robot left in the sand. An example of this can be seen in Figure 11, where there are actually two sets of virtually identical tracks heading to the left (along the marked arrows).

Post-analysis revealed that the final position errors during the sand field tests were in part due to the fact that odometry was slightly over-predicting distance travelled along the path (even in the lab on hard terrain). Odometry does affect the final visual estimate, particularly if the visual features being used are far away (e.g., horizon features). This slight miscalibration does not

account for the catastrophic pure-odometry estimation errors of Figure 11. This over-prediction effect was made less severe through further calibration.

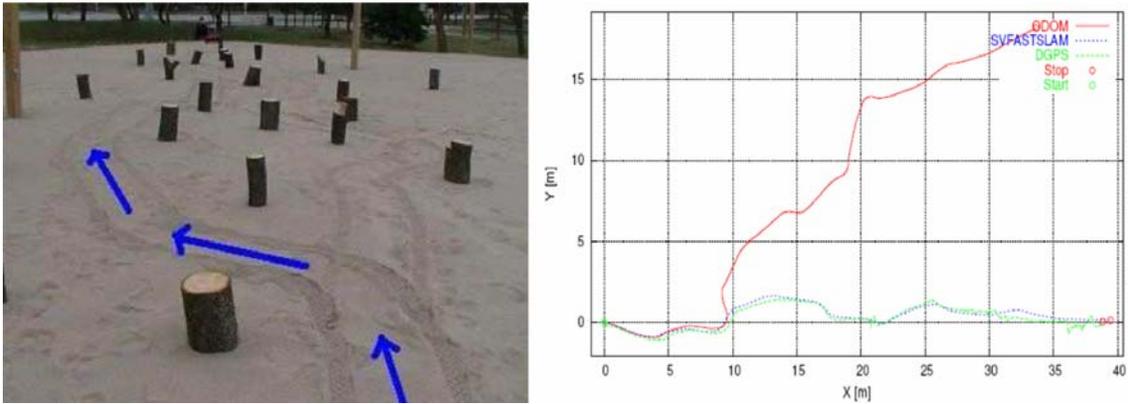


Figure 11: (above) Sand test site with arrows indicating path taken by rover. (below) Path estimated by FastSLAM (blue), odometry (red), and GPS (green) on loose sand.

#### Outdoors, Gravel:

A second set of field trials was conducted on a large gravel area (5 traverses total). Figure 12 shows the results of an approximately 120 m traverse at maximum speed of 10 cm/s on gravel. Here we found that odometry did not experience isolated positioning errors, as was the case on sand, but did experience a systematic error (gradual curve to the left). All of the trials at this site had similar errors, likely due to a slightly higher tire pressure on one side of the rover than the other. This systematic error was not observed prior to the field test; it was attributed to changes in tire pressure on the test day and hence miscalibration of odometry.

Using a tape measure, the final position of the rover was 117.4 m from the start. Visual motion estimation found 116.8 m and DGPS found 117.6 m. The tape measure was taken as ground truth, indicated the visual motion estimation under-predicted the position by 0.5% of distance travelled. Again, most of this positioning error was in the longitudinal direction (along the line joining the start and final positions). Visually, the lateral error was extremely small, indicating that orientation was estimated very well throughout the traverse.

The results of other trials at this site were mixed as we tried to push the system to move more quickly and use fewer images. Increasing the vehicle speed to 20 cm/s, or decreasing the frame-rate to 1.5 Hz, resulted in decreased performance.

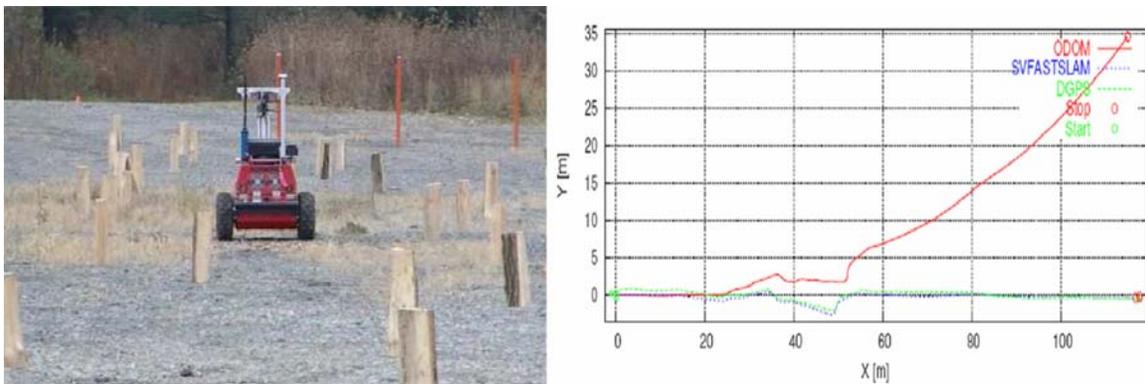
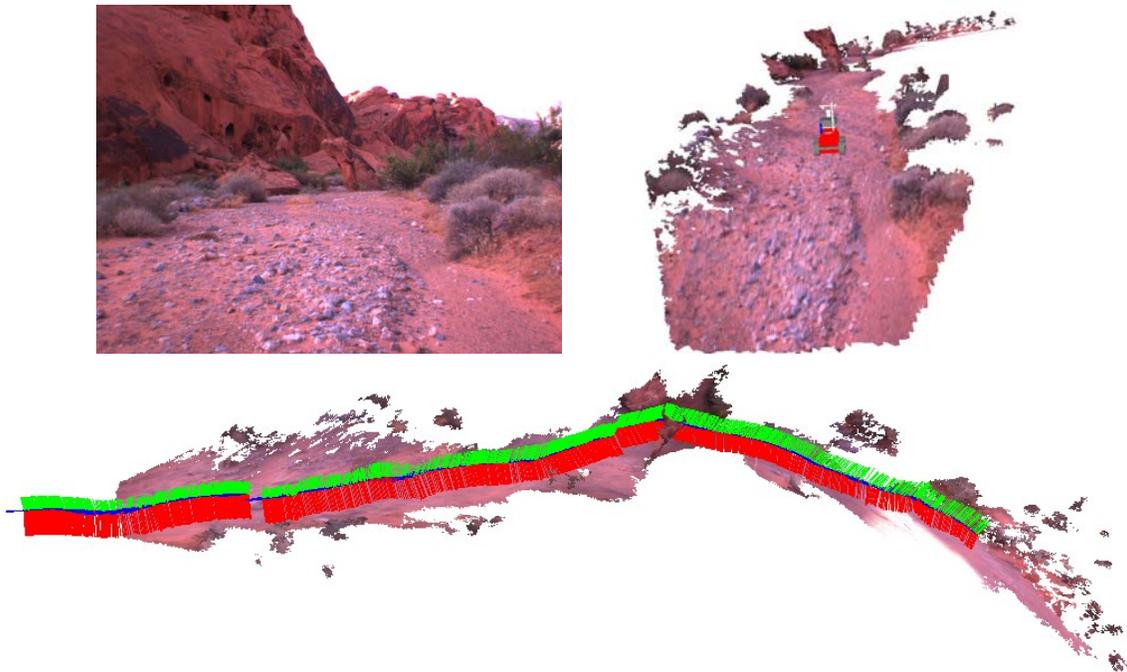


Figure 12: (above) Gravel test site. (below) Path estimated by FastSLAM (blue), odometry (red), and GPS (green) on loose sand.

## 5.2 Vision-Based Terrain Models

Figure 13 shows a model we created from a moving rover that traveled over 40m in a desert in Nevada. The first image in the sequence is shown for the right camera as well as the recovered camera motion, from our vision-based localization using SIFT features. A three dimensional model of the rover has also been inserted into the resulting terrain model for visualization.



*Figure 13: (left) First image in a stereo sequence captured in a desert in Nevada. (right) Terrain model with rover model inserted. (bottom) Resulting terrain model and motion of camera reference frame*

## 6. DISCUSSION

### 6.1 Related Work

There have been various works on visual motion estimation for planetary rovers with promising results. Semi-sparse terrain maps were constructed and matched successively to obtain a vision-based state estimate in [23]. An extended Kalman Filter was then applied to fuse with wheel odometry. Experiments at JPL's rover pit showed that the results had more than double the accuracy of the dead reckoning estimate.

A maximum likelihood estimation technique for rover localization in natural terrain was presented in [24] by matching range maps. Stereo vision generated local terrain range map which was matched to a previously generated 3D occupancy map to estimate rover pose. Good qualitative results were obtained when tested with Sojourner data, running on-board Rocky 7 Mars rover prototype.

Pixel tracking in stereo image sequences was proposed in [25] to estimate visual odometry in outdoor unstructured terrain, with around 4% error over 25 metres. [26] evaluated a similar algorithm on the Marsokhod robot on many runs totaling several hundreds of metres and achieved about 2% translation error. [27] proposed that a set of concurrent and complementary algorithms

are required for rover localization, as no single localization algorithm is robust enough to fulfill various localization needs during long range navigation.

In addition to stereo vision, [28] discussed the use of inertial sensors to estimate camera ego-motion and to augment stereo tracking on rough terrain. [12][29] showed that even with a robust stereo ego-motion method, the system accumulated super-linear error due to increasing orientation error. Therefore, they proposed incorporating an absolute orientation sensor to reduce the error growth to linear. They achieved 1.2% error in experiments carried out with a prototype Mars rover. This same technique was also used to perform rover path tracking [30].

The Mars Exploration Rovers, Spirit and Opportunity, have been using a derivative of this visual odometry technology on Mars [31]. Initial demonstrations were conducted on Spirit during Sols 175-178 with positive results. The technique was also used to improve odometry when Spirit was forced to drive using only five of its six wheels, although some problems were reported during Sols 416-421. An official report of the results is pending. The bundle adjustment method is also employed for rover localization, by taking a global approach in building an image network of the landing site [32]. The images were processed on Earth and incremental adjustments are uploaded every sol.

Realtime visual odometry results for terrestrial applications have also been reported by [33], which uses Harris corners for features. Our technique differs in that we are actually building a database of landmarks. Realtime SLAM results using SIFT features have recently been demonstrated indoors by [34][35] with a monocular camera and much smaller images than ours.

Most of the previous work used vision systems for localization only, whereas we also use the vision system for 3D modeling. Recently, [36] proposed using stereo images for recalibration and also for reconstructing 3D terrain models which were texture mapped with the original images. They have carried out preliminary experiments to create digital elevation maps at the ESA planetary terrain testbed. The model was then used to plan a trajectory for the Nanokhod rover. However, their vision system was part of the lander, not on-board of the rover. Therefore, the terrain map generated will be limited to the surroundings of the landing site only.

## **6.2 Advantages/Disadvantages of Approach**

The key advantages of our localization approach are:

- Stereo cameras and odometry are already baselined on most future rovers and as such no new sensors need be added to enable vision-based localization.
- Our approach does not rely on any artificial infrastructure to localize and hence can be used far away from a lander for long-range science missions.
- Highly distinctive SIFT features are used as visual landmarks, enabling the repeated identification of landmarks to be quite robust.
- A Simultaneous Localization and Mapping (SLAM) approach is used for motion estimation (as opposed to single-frame odometry). This allows landmarks to be tracked over an arbitrary number of images frames and thereby help reduce accumulation of error.
- A probabilistic algorithm is used to estimate the rover's pose from a large number of landmarks. Through this approach observations of landmarks both near by and on the horizon may be combined as we can account for the quality of each landmark individually.

The current disadvantages of our localization approach are:

- The extraction of SIFT features requires considerable computational effort. We have been addressing this through implementation of core vision components on FPGA in preparation for flight.

- Although our vision-based localization was formulated using a derivative of the FastSLAM 2.0 algorithm, we have found the use of more than a single particle to represent the rover's trajectory to be computationally too expensive. Our experimental results have shown with a single particle we may still achieve reasonable results.

The key advantages of our terrain mapping approach are:

- It can be seamlessly integrated with our vision-based localization technique and hence terrain models can be created while the rover is in motion (as compared to relying on a stationary camera).
- Through the use of an efficient voxel representation of the models, terrain maps for both visualization by ground operators and cost maps for autonomous operations can be generated.
- The resulting visualization models (with texture mapping) can be transmitted over a communication channel at a greatly reduced bandwidth than all of the raw image sources.

The key disadvantages of our terrain mapping approach are:

- The computational burden of generating disparity maps, voxel maps, and texturing is reasonable high and may require some steps to be implemented in hardware for flight operations.

### **6.3 Flight Readiness**

Our results to date are preliminary. The biggest technical obstacle to overcome in preparing for flight is keeping the computational burden within the expected envelope for future missions. We have begun to address this through the implementation of SIFT feature extraction on a FPGA, but it is likely that this idea will need to be extended to some of the other blocks in Figure 2 such as undistortion/rectification, stereo feature matching, disparity maps.

The visual odometry results of the Mars Exploration Rovers will hopefully provide enough evidence to baseline some form of vision-based localization on the next round of Mars missions. We have, for example, baselined vision-based localization in our preliminary design of the ExoMars rover shown in Figure 1. For our approach to be seriously considered it will be important to show our algorithms working on a representative avionics box, rover and natural terrain. We are working towards this end and feel it is realistic to expect the technology to be ready for a 2011 flight.

The operational risk of using a vision-based localization technique would seem to be reasonably low when compared to autonomous obstacle avoidance and route planning. Moreover, vision-based localization may actually reduce the risk of using some of these other computational intelligence technologies due to improved knowledge of rover location.

## **7. CONCLUSIONS AND FUTURE WORK**

We have demonstrated the ability for a rover to use a stereo camera and SIFT features as the landmarks for efficient localization and terrain mapping. The resulting online visual motion estimation was used for autonomous outdoor rover traverses up to 120 m long on loose terrain. The final positioning errors were 0.5% to 4% of distance travelled, a major improvement over using odometry alone. We plan to carry out more extensive experiments with the use of a surveyor's instrument to establish better ground truth in the future.

We have constructed terrain models in many environments, both artificial and natural including underground mines and buildings. We have also begun to address the computational requirements of our approach by implementing one of the core vision blocks on a FPGA. All of

these preliminary findings have shown promise and hence we continue to develop our vision technologies for planetary rovers.

In terms of our estimation algorithm, a future step in our work is to incorporate loop-closure detection and possibly backwards correction [37]. This could be incorporated in our outlier detection scheme as the number of outliers tends to spike when loops are closed. This is because a large number of SIFT matches are made but not in the expected locations. If loop closure can be robustly detected, we could switch to a ‘kidnapped robot’ scenario to reset the localization and make corrections backwards in time. Work must also be done to prune and rebalance the kd-tree to allow significantly longer operation of the algorithm. We also seek to make our approach more robust to the translation and rotation that can occur between consecutive images. Currently, we can tolerate only small translations and rotations. For practical applications we would like to be able to move 1 m in translation and 20 degrees in rotation. This would make the algorithm efficient enough for planetary exploration, where computational resources are scarce.

We are also currently building a prototype of the ExoMars rover design shown in Figure 1 that will be used to further test our vision-based localization and terrain modeling. We plan to conduct long-range (i.e., traverses of kilometers) field trials with this new rover during the summer of 2006 in a Mars-like environment (possibly Haughton Crater in the Canadian High Arctic).

## 8. ACKNOWLEDGEMENTS

We would like to thank Ho-Kong Ng, Kirk Harasym, Lucas Szajek, Tariq Rafique, Robert Nguyen, and Raymond Oung for their contributions and useful discussions. MDA, Space Missions would also like to thank David Lowe at the University of British Columbia for providing most of the code related to SIFT features under license. The Mars Exploration Rover images were provided courtesy of NASA/JPL-Caltech.

## REFERENCES

- [1] European Space Agency. Systems Requirements Documents, ExoMars Rover.
- [2] T. Parker, M. Malin, M. Golombek, T. Duxbury, A. Johnson, J. Guinn, T. McElrath, R. Kirk, B. Archinal, L. Soderblom, R. Li, and the MER Navigation Team and Athena Science Team. Localization, Localization, Localization. *Lunar and Planetary Science XXXV* (2004).
- [3] P. Goel, S.I. Roumeliotis, and G.S. Sukhatme. Robust Localization Using Relative and Absolute Position Estimates. In *Proceedings of the 1999 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS'99)*.
- [4] B. Wilcox and T. Nguyen. Sojourner on Mars and Lessons Learned for Future Planetary Rovers. In *Proceedings of the 28th International Conference on Environmental Systems (ICES'98)*, Danvers, MA, July 13-16, SAE publication 981695, Society of Automotive Engineers, Inc., 1998.
- [5] E.T. Baumgartner, H. Aghazarian, and A. Trebi-Ollennu. Rover Localization Results for the FIDO Rover. In *Proceedings of SPIE Vol. 4196, Sensor Fusion and Decentralized Control in Robotics System III*, 2000.
- [6] K. Iagnemma et al. Terrain estimation for enhanced autonomous rover mobility. In *Intelligence for Space Robotics*, 2006.
- [7] <http://anserver1.eprsl.wustl.edu/>
- [8] D.G. Lowe. Object recognition from local scale-invariant features. In *Proceedings of the Seventh International Conference on Computer Vision (ICCV'99)*, pages 1150–1157, Kerkyra, Greece, September 1999.

- [9] D.G. Lowe, Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, vol. 60, no. 2, pages 91-110, 2004.
- [10] C.J. Harris and M. Stephens. A combined corner and edge detector. In *Proceedings of 4th Alvey Vision Conference*, pages 147–151, Manchester, 1988.
- [11] S. Se, D. Lowe, and J. Little. Mobile robot localization and mapping with uncertainty using scale-invariant visual landmarks. *International Journal of Robotics Research*, vol. 21, no. 8, pages 735–758, August 2002.
- [12] C.F. Olson, L.H. Matthies, M. Schoppers, and M.W. Maimone. Robust stereo ego-motion for long distance navigation. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Volume 2*, pages 453–458, South Carolina, June 2000.
- [13] J.S. Beis and D.G. Lowe. Indexing without invariants in 3D object recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 21, no. 10, pp. 1000–1015, 1999.
- [14] M. Montemerlo, S. Thrun, D. Koller, and B. Wegbreit. Fastslam 2.0: An improved particle filtering algorithm for simultaneous localization and mapping that provably converges. In *Proceedings of International Joint Conference on Artificial Intelligence (IJCAI)*, Acapulco, Mexico, August 9-15 2003.
- [15] M. Montemerlo. Fastslam: A factored solution to the simultaneous localization and mapping problem. Ph.D. dissertation, Robotics Institute, Carnegie Mellon University, Pittsburgh, PA, July 2003.
- [16] T. Barfoot. Online Visual Motion Estimation using FastSLAM with SIFT Features. In *Proceedings of International Conference on Robotics and Intelligent Systems (IROS)*, Edmonton, Alberta, August 2-6, 2005.
- [17] R.E. Kalman. A new approach to linear filtering and prediction problems. *Trans. ASME, J. of Basic Eng.*, vol. 82, pages 35-45, 1960.
- [18] M.A. Fischler and R.C. Bolles. Random sample consensus: a paradigm for model fitting with application to image analysis and automated cartography. *Communications of ACM*, vol. 24, pages 381-395, 1981.
- [19] E. Kraft. A quaternion-based unscented kalman filter for orientation tracking. In *Proceedings of the 6th International Conference on Information Fusion (ISIF)*, Cairns, Australia, July 2003.
- [20] D. Scharstein and R. Szeliski. A taxonomy and evaluation of dense two-frame stereo correspondence algorithms. *International Journal of Computer Vision*, vol. 47, 2002.
- [21] R. Zabih and J. Woodfill. Non-parametric local transforms for computing visual correspondence. In *Proceedings of European Conference on Computer Vision (ECCV) Volume 2*, pages 151–158, Stockholm, Sweden, 1994.
- [22] G. Roth and E. Wibowo. An efficient volumetric method for building closed triangular meshes from 3-d image and point data. In *Proceedings of Graphics Interface (GI)*, pages 173–180, Kelowna, B.C., Canada, 1997.
- [23] B. Hoffman, E.T. Baumgartner, T.L. Huntsberger, and P.S. Schenker. Improved rover state estimation in challenging terrain. *Autonomous Robots*, vol. 6, pages 113–130, 1999.
- [24] C.F. Olson and L.H. Matthies. Maximum likelihood rover localization by matching range maps. In *Proceedings of International Conference on Robotics and Automation*, pages 272–277, Leuven, Belgium, May 1998.
- [25] A. Mallet, S. Lacroix, and L. Gallo. Position estimation in outdoor environment using pixel tracking and stereovision. In *Proceedings of IEEE International Conference on Robotics and Automation (ICRA)*, pages 3519–3524, San Francisco, April 2000.
- [26] S. Lacroix, A. Mallet, D. Bonnafous, G. Bauzil, S. Fleury, M. Herrb, and R. Chatila. Autonomous rover navigation on unknown terrains: functions and integration. *International Journal of Robotics Research*, vol. 21, no. 10–11, pages 917–942, October-November 2002.

- [27] S. Lacroix and A. Mallet. Integration of concurrent localization algorithms for a planetary rover. In Proceedings of International Symposium on Artificial Intelligence and Robotics and Automation in Space (iSAIRAS), Canadian Space Agency, Quebec, Canada, June 2001.
- [28] K. Nickels and E. Huber. Inertially assisted stereo tracking for an outdoor rover. In Proceedings of IEEE International Conference on Robotics and Automation (ICRA), pages 3078–3083, Seoul, Korea, May 2001.
- [29] C.F. Olson, L.H. Matthies, M. Schoppers, and M.W. Maimone, Rover navigation using stereo ego-motion. *Robotics and Autonomous Systems*, vol. 43, no. 4, pp. 215-229, 2003.
- [30] D.M. Helmick, Y. Cheng, D.S. Clouse, L.H. Matthies, S.I. Roumeliotis. Path following using visual odometry for a Mars rover in high-slip environments. In Proceedings IEEE AERO 2004.
- [31] M. Maimone et al. Surface navigation and mobility intelligence on the Mars Exploration Rovers. In *Intelligence for Space Robotics*, 2006.
- [32] R. Li, K. Di, L.H. Matthies, R.E. Arvidson, W.M. Folkner, B.A. Archinal. Rover localization and landing site mapping technology for the 2003 Mars Exploration Rover missions. *Journal of Photogrammetry Engineering and Remote Sensing*, vol. 70, no. 1, pages 77-90, 2003.
- [33] D. Nister, O. Naroditsky, and J. Bergen. Visual odometry. In Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR), Washington, DC, June 2004.
- [34] L. Goncalvez, E. Di Bernardo, D. Benson, M. Svedman, J. Ostrowski, N. Karlsson, P. Pirjanian, A Visual Front-end for Simultaneous Localization and Mapping. In Proceedings of IEEE International Conference on Robotics and Automation (ICRA), Barcelona, Spain, 2005.
- [35] N. Karlsson and E. Di Bernardo. The vSLAM Algorithm for Robust Localization and Mapping. In Proceedings of IEEE International Conference on Robotics and Automation (ICRA), Barcelona, Spain, 2005.
- [36] M. Vergauwen, M. Pollefeys, and L. Van Gool. A stereo-vision system for support of planetary surface exploration. *Machine Vision and Applications*, vol. 14, pages 5–14, 2003.
- [37] S. Se, D.G. Lowe and J.J. Little, Vision-based Global Localization and Mapping for Mobile Robots. *IEEE Transactions on Robotics*, vol. 21, no. 3, pages 364-375, June 2005.