Terrain Mapping for a Roving Planetary Explorer

M. Hebert, C. Caillas, E. Krotkov, I. S. Kweon, T. Kanade [1]
The Robotics Institute
Carnegie Mellon University
Pittsburgh, PA 15213

Abstract

We are prototyping a legged vehicle for an exploratory mission on another planet, conceivably Mars, where it is to traverse uncharted areas and collect material samples. This paper describes how the rover can construct from range imagery a geometric terrain representation — an elevation map that includes uncertainty, unknown areas, and local features. First, it presents a new algorithm to construct an elevation map from a single range image. By virtue of working in spherical-polar space, the algorithm is independent of the desired map resolution and the orientation of the sensor, unlike other algorithms that work in Cartesian space. Second, it presents a novel two-stage matching technique (feature matching followed by iconic matching) to identify the transformation T corresponding to the vehicle displacement between two viewing positions. Third, to support legged locomotion over rough terrain, it describes new methods to evaluate regions of the constructed elevation maps as footholds.

1 Introduction

We are prototyping a legged vehicle called the Ambler (fig. 1) for an exploratory mission on another planet, conceivably Mars, where it is to traverse uncharted areas and collect material samples. Planetary exploration poses significant challenges for rovers: unprecedented levels of autonomy and reliability due to communication delays that limit conventional Earth-based teleoperation; and traversal of rugged, irregular terrain for which existing mechanisms and perception techniques are inadequate.

Papers that describe the background of our work include a comprehensive account of the Ambler configuration [1] and an overview of the integrated research program [4]. The aim of this paper is to describe first results from the Ambler perception system.

The Ambler perception system must build and maintain representations of the terrain and discrete objects—*terrain maps* that are appropriate for a wide variety of tasks, each with different requirements. For example, locomotion and sampling require detailed, local representations, while navigation and mission planning demand broad, global descriptions. In this paper, we do not address the full scope of the perception system; we focus only on building maps based on the observations of a single sensor, and using those maps to support locomotion.

This paper addreses sensing in section 2, and presents a new technique for constructing elevation maps in section 3. It presents a two-stage matching technique in section 4, as well as a rule for combining overlapping maps. It describes methods for analyzing map geometry for locomotion in section 5, and documents experimental methods and results in section 6. It concludes by discussing limitiations and future work.

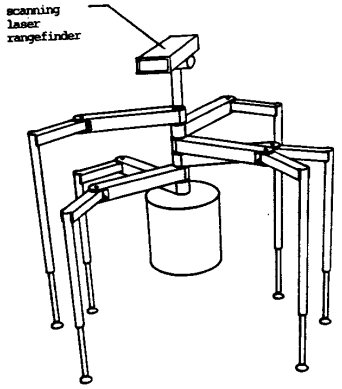

Figure 1: The Ambler

2 Active Range Sensing

The Ambler perception system will use multiple sensing modalities, both imaging and non-imaging. Here we concentrate on active range sensors, which measure the distance to an object in the environment by observing the reflection of a reference signal (sonar, laser, radar, etc.) from the object. Active sensors offer two chief advantages: they provide range data without the numerous computations required by passive techniques such as stereo vision; and they are largely insensitive to illumination conditions, thus simplifying the image analysis problem (which is especially important for images of outdoor scenes in which illumination can be neither controlled nor predicted).

We use the ERIM scanning laser range finder, which measures the phase difference between an amplitude-modulated laser beam and its reflection from a point in the scene [9]. We measure the coordinates of the point in a non standard spherical polar reference frame, in which ρ is the measured range, and ϕ and θ are the vertical and horizontal scanning angles of the beam direction corresponding to row and column position in the image.

[1]This research was sponsored by NASA under Contract NAGW-1175. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of NASA or the U.S. Government.

The Cartesian coordinates of a point measured in spherical polar coordinates have been derived [3] as

$$x = \rho \sin\theta \quad , \quad y = \rho \cos\phi \cos\theta \quad , \quad z = \rho \sin\phi \cos\theta \quad . \qquad (1)$$

3 Constructing Elevation Maps

Applying eq. 1 to the measurements in a range image yields an elevation map. However, this map is non-uniform in Cartesian space, because the coordinate transformation is non-linear. Further, the map grows less dense and less accurate with increasing distance from the sensor.

One could circumvent the former difficulty by using a map structure that is not a regularly spaced grid, such as a Delaunay triangulation. However, this is not practical because of the complex algorithms required to access data points and their neighborhoods. Another approach is to interpolate between data points to build a dense elevation map on a grid, either by approximating the surface between data points (e.g., as a bicubic surface), or by globally fitting a surface under some smoothness assumptions (e.g., regularization). However, both of these approaches have signficant limitations: they make assumptions on the local shape of the terrain which may not be valid in the case of rough terrain; and they depend heavily on the resolution and position of the grid (i.e., they cannot compute an estimate of the elevation at an (x, y) position that is not a grid point without resampling the grid).

We propose an alternative, the locus algorithm, that uses a model of the sensor to interpolate at *arbitrary resolution* without making any assumptions on the terrain shape other than the continuity of the surface.

3.1 Locus Algorithm

The problem of finding the elevation z of a point (x, y) is equivalent to computing the intersection of the surface observed by the sensor with the vertical line passing through (x, y). The basic idea of the locus algorithm is to convert the latter formulation into a problem in image space (specifically, spherical-polar space rather than row-column space, fig. 2). A vertical line[2] is a locus (curve) in image space, whose equation as a function of ϕ is derived by inverting eq. 1, assuming x and y constant:

$$\rho = \rho_l(\phi) = \sqrt{\frac{y^2}{\cos^2\phi} + x^2} \quad , \quad \theta = \theta_l(\phi) = \arctan\frac{x\cos\phi}{y} \quad . \qquad (2)$$

Similarly, the range image can be viewed as a surface $\rho = I(\phi, \theta)$ in $(\phi - \theta)$ space. The problem then is to find the intersection, if it exists, between a curve parameterized by ϕ and a discrete surface. Since the surface is known only from a sample of data, the intersection cannot be computed analytically.

Instead, we must search along the curve for the intersection point. Let $\hat{\theta}_l(\phi)$ be the image column closest to $\theta_l(\phi)$, and let $\Delta(\phi_j) \equiv \rho_l(\phi_j) - I(\phi_j, \hat{\theta}_l(\phi_j))$. The search proceeds in two stages. First, we locate the two scanlines of the range image, ϕ_1 and ϕ_2, between which the intersection must be located, i.e., such that $\mathrm{sgn}\Delta(\phi_1) \neq \mathrm{sgn}\Delta(\phi_2)$. Second, we apply a binary search between ϕ_1 and ϕ_2. The search stops when $|\phi_n - \phi_{n+1}| < \epsilon$ (i.e., the resolution of the elevation is controlled by the parameter ϵ). Third, since there are no pixels between ϕ_1 and ϕ_2, we perform Lagrangian interpolation for $\phi_1 < \phi < \phi_2$, using as control points the four pixels that surround the intersection point. The result is a value ϕ that is mapped to ρ and θ by eq. 2, and then mapped to an elevation value by eq. 1. Repeating this for vertical lines at every desired (x, y) point yields a dense elevation map of the desired resolution, as required.

[2]We have generalized the locus algorithm from the case of a vertical line to the case of a general line in space [3], which allows us to build maps using any reference plane, not just the xy plane. We present the case of the vertical line to simplify exposition.

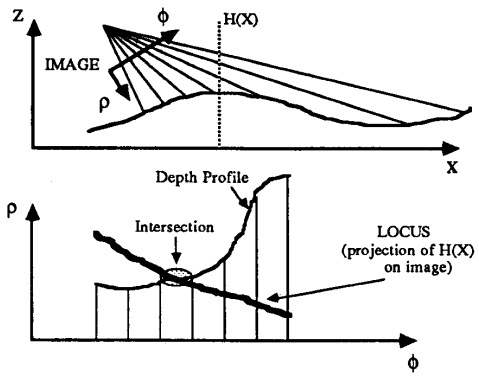

Figure 2: Imaging geometry (top), 1-D locus (bottom)

3.2 Range Shadows

Objects in the environment may cast range shadows (cause occlusions). It is important to identify the occluded regions, because if we apply the locus algorithm there directly, then the surface would be smoothly interpolated, possibly incorrectly. In turn, this could lead the rover to plan a path through that region, expecting it to be traversable when in fact it is unknown.

One could detect empty regions in the elevation map given by eq. 1, without interpolation. This does not work, because the size of the shadow regions may be on the order of the average distance between data points (this is especially true for distant regions in which the distribution of data points is sparse).

Another approach is to incorporate the detection of shadow regions into the locus algorithm, again working in image space. We observe that a range shadow corresponds to an occluding edge in the image. As in fig. 3, an (x, y) location in the map is in a shadow area if its locus intersects the image at a pixel that lies on such an edge. We implement this idea by first detecting edges in the range image by using the GNC algorithm [2]. Then, when we apply the locus algorithm and observe that the locus of a given location intersects the image at an edge pixel, we mark that location as lying in a range shadow.

3.3 Uncertainty

We have developed a probabilistic model of the uncertainty on the sensor measurements, according to which the measured range errors are normally distributed with standard deviation propor-

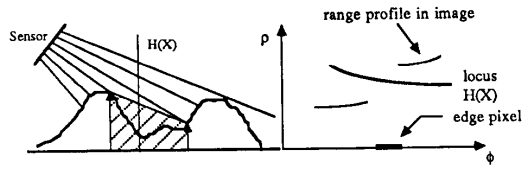

Figure 3: Shadowed area (left), discontinuity (right)

tional to the square of measured range ([3], p. 7). The range measurement uncertainty is oriented along the direction of measurement (fig. 4).

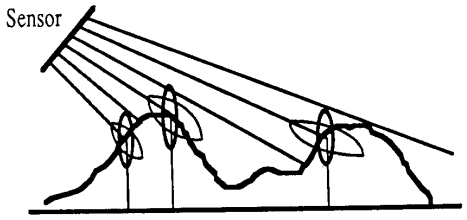

Figure 4: 1-D uncertainties on sensor and map

To identify the uncertainty on the elevation value at each grid point (x, y), as part of the locus algorithm we transform the uncertainty on a sensor measurement so that it is oriented along the z axis ([3], pp. 25-27). This conversion is non trivial, since the the range uncertainty is distributed across a region in the elevation map. According to this model, the distribution of elevation errors is approximately normal, with standard deviation proportional to the product of measured range and elevation.

4 Combining Elevation Maps

We have so far addressed the problem of building a representation of the environment from sensor data collected at one fixed location. But over the course of a mission, the rover must deal with a stream of images. Processing multiple views yields at least two benefits. First, by identifying the transformation between viewing positions the perception system can independently estimate the vehicle displacement. Second, merging maps into a composite can a) increase the resolution of the parts of indiviual elevation maps originally measured at a distance from the vehicle, and b) add information about previously occluded areas.

We propose a two-stage approach, using feature matching and iconic matching together. The former has been studied extensively; the latter has been successfully applied to incremental depth estimation [5] and map matching [8]; to our knowledge, they have not been used together. The result is an estimate T of the transformation between the two views. We then apply T to each of the maps, and merge them into one common, composite elevation map.

4.1 Feature matching

Given an initial estimate T_0 of the transformation, and two elevation maps, we seek a better estimate T_F of the transformation by extracting three-dimensional features from the elvation maps, and then identifying the correspondences between them.

Feature extraction We extract from each elevation map a set of three-dimensional features, F_i^1 and F_j^2. This begins by extracting points of high surface curvature, based on the magnitude of the two principal curvatures of the elevation map surface. Then we group the extracted points, and classify each group as a point, line segment, or region, according to its size, elongation, and curvature distribution.

Feature correspondence We identify the correspondence between the features extracted from elevation maps 1 and 2 using the predict-and-verify paradigm. The best correspondence determines the transformation T_F such that $F_{jk}^2 \approx T_F(F_{ik}^1)$.

We identify (predict) candidate matches based on the similarity of the length of the lines and the similarity of the curvature strength of the points. Each candidate is a set of pairs $S_k = (F_{ik}^1, F_{jk}^2)$, and each of the S_k induces a rigid transformation T_{Fk}.

We evaluate (verify) each candidate transformation T_{Fk} by computing the distance D between the features:

$$D = \sum_k d(F_{ik}^1 \ , \ T_{Fk}(F_{ik}^2)) \ , \tag{3}$$

where the distance $d(\cdot)$ depends on the feature type. The search in the space of candidate matches begins with $T_{Fk} = T_0$, and continues in a depth-first fashion, considering all of the pairings S_k and transformations T_{Fk}. The pairing minimizing D, say S_K, is the final match between the two maps, and $T_F \leftarrow T_{FK}$ is the best transformation.

4.2 Iconic matching

Given the estimate T_F and a pair of range images, the iconic matching algorithm seeks the transformation T_I that minimizes an error function computed not at distinct features, but over all measurements. The idea is to iteratively adjust the transformation parameters so as to minimize the error function, taking into account all of the data.

1) Compute map from image 1 According to the formulation of the locus algorithm, a point in the i^{th} elevation map lies at the intersection of a line containing this point (e.g., the line $H(x)$ in fig. 2) with a locus of points projected onto the i^{th} range image. Let (u, v) be that line, parameterized respectively by a point and a unit vector.[3] Let $f_i(u, v)$ be the function that maps the line to a point in the i^{th} elevation map (this function is computed by the generalized locus algorithm). So the first step of the iconic matching algorithm is to compute $f_1(u, v)$, for which the lines (u, v) are vertical.

[3]The (u, v) representation is not the minimal representation of lines in space [7]. We have implemented our algorithms with the non-redundant (a, b, p, q) representation [3], but for clarity of exposition we will utilize the (u, v) representation.

2) Gradient descent. The iconic matching algorithm now searches for the "best" transformation parameters by iterative gradient descent on an error function E (to be defined below). Let $\nu = [\alpha, \beta, \gamma, t_x, t_y, t_z]$ represent the transformation parameters, where the first three are the rotation angles and the last three comprise the translation vector. The error E reaches a minimum when $\frac{\partial E}{\partial \nu} = 0$. Assuming a reasonably accurate initial estimate of ν by T_F, the minimum error can be achieved by an iterative gradient descent of the form

$$\nu^{i+1} = \nu^i + k\frac{\partial E}{\partial \nu}(\nu^i) , \tag{4}$$

where ν^i is the estimate of ν at iteration i.

At each iteration, the algorithm a) applies the updated transformation to the measurements in range image 2, b) computes the error between the first and second (transformed) measurements, and c) updates the transformation parameters. Iteration continues until either the variation of error ΔE is small enough (convergence), or E itself is small enough (smaller than what can be reasonably achieved given the characteristics of the sensor).

2a) Apply transformation In order to evaluate the estimated transformation, we must first apply it to the second range image or to the second elevation map. We will employ the generalized locus algorithm to accomplish this.

The function $f_2(u, v)$ computes an elevation map from range image 2. If $(u', v') = (Ru + t, Rv)$ is the line (u, v) in elevation map 1, transformed by T_I, then $f_2(u', v')$ represents the elevations in map 2 of the same grid locations at which elevation have been computed in map 1. (Incidentally, this demonstrates one important advantage of the generalized locus algorithm: even though the transformed line (u', v') can be anywhere in the coordinate system of image 2 — and in general is not parallel to the map z axis — the locus algorithm computes $f_2(u', v')$ directly, without interpolation or resampling.) Now if we refer both elevation estimates to the same frame, say by applying the inverse of T_I to $f_2(u', v')$, then we can compare the two directly.

To distill this analysis into a single equation, let $g(u, v, T_I)$ be the elevation in map 2 at the "location" (u, v), referred to map 1:

$$g(u, v, T_I) = T_I^{-1}(f_2(u', v')) = R' f_2(u', v') + t' , \tag{5}$$

where $T_I^{-1} = (R', t') = (R^{-1}, -R^{-1}t)$, and $(u', v') = (Ru + t, Rv)$.

2b) Compute error To determine the quality of the match between the measurements referred to a common coordinate frame by step 2a, we take the error function to be

$$E = \sum \|f_1(u, v) - g(u, v, T_I)\|^2 , \tag{6}$$

where the summation is taken over all (u, v) where both $f_1(u, v)$ and $g(u, v, T_I)$ are defined. In words, E is the sum of the squared differences between the elevation at a location in the first map and the elevation at the same location computed from the second map using T_I. Technically, E should be divided by the number N of common points, since the overlap region between the two maps is not known in advance. However, we empirically observe that N does not vary significantly between iterations, so for now we neglect to normalize.

2c) Update transformation parameters The update rule is given by eq. 4. What remains to be determined in that equation is the partial derivative of the error function with respect to each of the transformation parameters.

From eq. 6, the derivative of E is

$$\frac{\partial E}{\partial \nu} = -2\sum (f_1(u, v) - g(u, v, T_I))\frac{\partial g}{\partial \nu}(u, v, T_I) . \tag{7}$$

From eq. 5, the derivative of g is

$$\frac{\partial g}{\partial \nu}(u, v, T_I) = R'\frac{\partial f_2}{\partial \nu}(u', v') + \frac{\partial R'}{\partial \nu} f_2(u', v') + \frac{\partial t'}{\partial \nu} . \tag{8}$$

We can compute analytically the derivatives appearing in the last two terms in eq. 8. To compute the derivative of $f_2(u', v')$ with respect to ν, we apply the chain rule and compute the derivative with respect to each component of ν. This completes the identification of $\frac{\partial E}{\partial \nu}$, and thus, of the new transformation parameters ν^{i+1}.

4.3 Combination Rule

Once the matching algorithms identify the transformation corresponding to the displacement between two viewing positions, we apply it pairwise to the sequence of images, producing a single, composite elevation map as follows. We simply add non-overlapping points to the composite map. For overlapping points, we replace $f_1(u, v)$ by the maximum likelihood estimate

$$f_1(u, v) \leftarrow \frac{\sigma_2^2 f_1 + \sigma_1^2 f_2}{\sigma_1^2 + \sigma_2^2} , \tag{9}$$

where σ_1 and σ_2 are the standard deviations of the uncertainty distributions on the two elevation estimates.

5 Evaluating Footfalls

A perceptual task unique to legged locomotion is to evaluate terrain regions as footfall locations (foot placements). This is essential for locomotion over the rugged terrain that could be encountered on the surface of other planets such as Mars. In this section, we describe several methods to evaluate elevation map regions as footfall locations. These methods operate on the geometric structure of the surface described by the elevation map, for now ignoring important material properties of the soil such as load-bearing strength, compliance, and coefficient of friction. While incomplete, these methods are considerably more sophisticated than others reported in the literature, which require operator interaction [6].

An Ambler foot is modeled by a flat disk 30 cm in diameter. The problem is to find the "best" foot-shaped subregion B in a given region R of a given elevation map (R is computed elsewhere based on the current heading and gait). We have developed five solutions, corresponding to different measures of "best," and present them in increasing order of sophistication.

Max-min Find B that minimizes the difference $z_{max} - z_{min}$ of extremal elevations, There are cases where this method prefers a flat surface punctuated by a single spike rather than an undulating surface. This is obviously undesirable.

Planar fit Find B that best fits a plane, subject to the constraint that the plane normal is approximately parallel to the leg. This method suffers the same deficiency as above.

Support area Find **B** that minimizes the depth of penetration d_{opt} into the soil required to achieve the minimum necessary support area A_{min} (contact area between foot and terrain, or the number of map points within the circumference of the disk that are above the plane of the foot). This method is superior to the previous two to the extent that it better accounts for the shape of the terrain. However, there are cases that it fails to distinguish, e.g., two sinusoidal surfaces with the same frequency but different amplitudes. This method should, but does not, select the surface with smaller amplitude variations.

Free volume Find **B** that minimizes the free (unoccupied) volume between the foot and soil, $V \equiv Nz_{max} - \sum_{i=1}^{N} z_i$. This method correctly discriminates the two sinusoidal surfaces described above. However, it does not take into account the distribution of "holes" in the surface or the consequences of applying force to (stepping on) the surface.

Equilibrium Find **B** that minimizes V and E, where $E \equiv \sqrt{m_x^2 + m_y^2}$ is the first moment of the mass distribution about the center of the foot, and $m_x = \sum_{i=1}^{N} x_i(z_{max} - z_i)$. The second condition ensures the footfall of greatest "equilibrium" (balance) with respect to holes in the surface. The idea is that as the foot contacts sandy soil, the sand fills the holes with a minimum of foot penetration into the soil, and as the foot contacts rocky soil, it exerts the minimum lateral forces on potentially unstable materials.

6 Experiments

First, we evaluated the locus algorithm on synthesized range images with additive Gaussian noise by comparing its performance to that of Cartesian space interpolation algorithms (cf. section 3). The results show that the locus algorithm is more stable with respect to surface orientation and noise level than the others ([3], p. 25). We conclude that this is due to performing the interpolation in image space instead of first applying eq. 1 to the data points.

Second, we tested the locus algorithm on a variety of real range images of uneven terrain. Qualitatively, the algorithm produces elevation maps that capture the structure of the terrain, correctly identifies shadowed regions, and computes meaningful uncertainty estimates that increase with distance. We have not quantitatively examined the accuracy of these elevation maps, since we lack ground truth elevation data.

Third, we tested our two-stage approach to combining elevation maps pairwise on a sequence of real range images of rugged terrain. In the first stage, feature matching provides an initial estimate T_F of the displacement between consecutive maps; in the second stage, iconic matching starts with T_F and estimates the best transformation T_I. We then apply T_I pairwise to merge the sequence of maps into a composite.

Figure 5 shows the result of feature matching in a case where a significant displacement separates two maps (rotation by about 30°, translation by about 2 m). The top image superimposes the two maps' features after applying T_F to one map; note that the linear features agree closely. The bottom image shows the correspondences between the map features, where the lower left shows the area common to both maps after applying T_F.

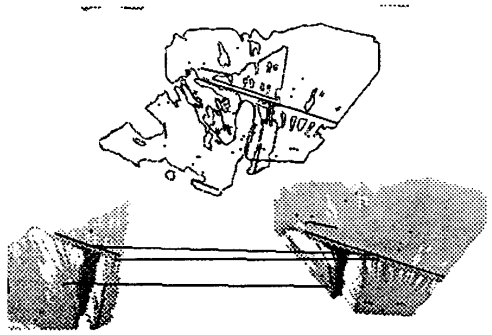

Figure 5: Feature matching results

Next, the iconic matching algorithm starts with T_F and computes T_I. For the pair of maps shown in fig. 5, the algorithm converges after approximately 30 iterations. The number of iterations required proves to vary considerably between different data sets. In practice, we find that T_F is usually sufficiently accurate to ensure convergence of the gradient descent algorithm to the global minimum. During testing with synthetic images we observed convergence to the global minimum despite errors in the initial estimate exceeding 1.5 m in translation and 10° in rotation.

Figure 6 shows the result of combining four elevation maps as an isoplot surface at 10 cm resolution. Using one of the maps (chosen arbitrarily) as ground truth, the RMS elevation error of the composite elevation map (approx. 6 cm/pixel) is commensurate to the resolution of the laser scanner (approx. 8 cm/pixel).

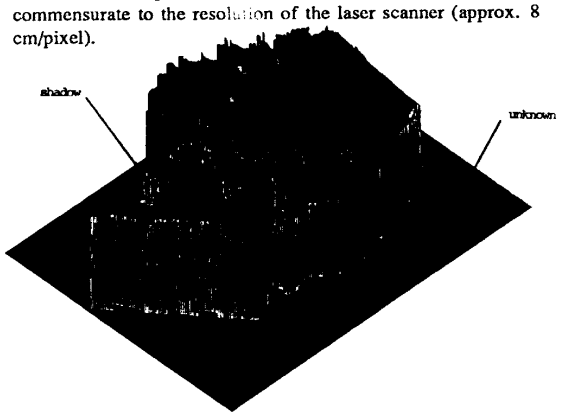

Figure 6: Four elevation maps combined

Finally, we tested a partially integrated Ambler system at an experimental testbed (fig. 7): a single leg with a fully operational controller; the range finder mounted above the leg; and a 25 m^2 "sandbox" of terrain to be traversed. The perception system communicates with other modules through queries, which typically are requests for the elevation map at a given resolution within a polygonal region. We evaluate the selected footfall location by servoing the leg there, thus closing the loop between perception and action. Visual inspection of the servoed positions

shows the selected locations to be reasonably accurate; quantitative error measurements are not yet available. Dozens of trials on different terrains suggest that the perception algorithms provide reliable and reasonably accurate descriptions of the terrain that suffice for moving the leg and executing footfalls.

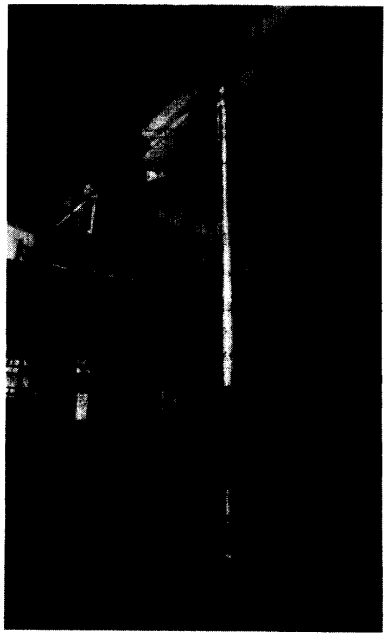

Figure 7: Single leg testbed

7 Discussion

In this paper we presented techniques to build terrain maps based on the observations of a single range sensor, and to use those maps to support locomotion: a new algorithm to build elevation maps at arbitrary resolution, including elevation uncertainty and unknown areas; a new two-stage matching algorithm to identify vehicle displacement between frames and construct composite elevation maps; and new methods for geometrically evaluating areas of the composite elevation map as footfall locations.

Preliminary experiments demonstrate that an integrated system can build and use maps to select footfall locations. This illustrates the advantages of working in image space rather than in Cartesian space.

While the first results are encouraging, further work is required both in map building and map analysis. For the former, we must complete an automatic calibration procedure to more accurately relate sensor and vehicle coordinate systems, incorporate elevation uncertainties in map matching (as distinct from map merging, where the maximum likelihood estimate includes uncertainty), and formally study the convergence properties of the iconic matching algorithm. For the latter, we must investigate more sophisticated footfall evaluations that take into account not only the local geometry of the terrain, but also geometric uncertainty and material properties of the soil such as load-bearing strength, compliance, and coefficient of friction. Further, we must better integrate the algorithms into the Ambler system, and make more quantitative assessments of their performance.

The work reported in this paper addresses a small fraction of the problems faced in developing a complete perception system for the Ambler. The scope of future research includes two broad categories: navigation and sampling. For the former, we aim to increase map coverage by processing multiple views from multiple sensors, and to determine vehicle position by landmark triangulation. For the latter, we intend to use surface topography to identify promising sample sites, and to build models of discrete objects both to select particular samples and to guide sample acquisition.

References

[1] J. Bares and W. Whittaker. Configuration of an Autonomous Robot for Mars Exploration. In *Proc. World Robotics Conference*, Society of Mechanical Engineers, Gaithersburg, Maryland, To appear, May 1989.

[2] A. Blake and A. Zisserman. *Visual Reconstruction*. MIT Press, Cambridge, Massachusetts, 1987.

[3] M. Hebert, T. Kanade, and I. Kweon. *3-D Vision Techniques for Autonomous Vehicles*. Technical Report CMU-RI-TR-88-12, The Robotics Institute, Carnegie Mellon University, 1988.

[4] E. Krotkov, J. Bares, M. Hebert, T. Kanade, T. Mitchell, R. Simmons, and W. Whittaker. An Autonomous Rover for Planetary Exploration. *IEEE Computer*, To appear, June 1989.

[5] L. Matthies and S. Shafer. Error modeling in stereo navigation. *IEEE Journal of Robotics and Automation*, 3:239–248, June 1987.

[6] F. Ozguner, S. J. Tsai, and R. B. McGhee. An Approach to the Use of Terrain-Preview Information in Rough-Terrain Locomotion by a Hexapod Walking Machine. *International Journal of Robotics Research*, 3(2):134–146, Summer 1984.

[7] K. S. Roberts. A New Representation for a Line. In *Proc. Computer Vision and Patter Recognition*, Ann Arbor, MI, 1988.

[8] R. Szeliski. Estimating Motion from Sparse Range Data without Correspondence. In *International Conference on Computer Vision*, Tarpon Springs, Florida, December 1988.

[9] D. Zuk, F. Pont, R. Franklin, and V. Larrowe. *A System for Autonomous Land Navigation*. Technical Report IR-85-540, Environmental Research Institute of Michigan, Ann Arbor, Michigan, 1985.