High-Resolution Terrain Map from Multiple Sensor Data

In So Kweon and Takeo Kanade

***Abstract*—This paper presents 3-D vision techniques for incrementally building an accurate 3-D representation of rugged terrain using multiple sensors. We have developed the *locus method* to model the rugged terrain. The locus method exploits sensor geometry to efficiently build a terrain representation from multiple sensor data.**

Incrementally modeling the terrain from a sequence of range images requires an accurate estimate of motion between successive images. In rugged terrain, estimating motion accurately is difficult because of occlusions and irregularities. We show how to extend the locus method to pixel-based terrain matching, called the *iconic matching method* to solve these problems. To achieve the required accuracy in the motion estimate, our terrain matching method combines feature matching, iconic matching, and inertial navigation data.

Over a long distance of robot motion, it is difficult to avoid error accumulation in a composite terrain map that is the result of only local observations. However, a prior digital elevation map (DEM) can reduce this error accumulation if we estimate the vehicle position in the DEM. We apply the locus method to estimate the vehicle position in the DEM by matching a sequence of range images with the DEM.

Experimental results from large-scale real and synthetic terrains demonstrate the feasibility and power of our 3-D mapping techniques for rugged terrain. In real world experiments, we built a composite terrain map by merging 125 real range images over a distance of 100 m. Using synthetic range images, we produced a composite map of 150 m from 159 images.

In this work, we demonstrate a 3-D vision system for modeling rugged terrain. With this system, mobile robots operating in rugged environments can build accurate terrain models from multiple sensor data.

***Index Terms*— Autonomous robots, matching, range images, rugged terrain, sensor fusion, terrain maps, 3-D vision.**

I. Introduction

Robots operating in rugged terrain need to plan their actions, safeguard their motions, and estimate their positions in both local and global contexts. *A priori* maps and plans are insufficient for navigating natural terrain because robots in real environments must cope with contingencies and respond to changes: Robots require accurate and reliable geometric terrain representations that are constantly updated with current terrain data.

Sensors like range finders provide geometric information about environments but not in a form needed for planning, safeguarding, and navigating. For instance, 3-D range images acquired by a laser scanner are given with respect to the sensor coordinate frame, but robots must plan and operate in another coordinate frame: the surface of the world. Traditional Cartesian-based approaches use the geometric relationship between the sensor and the world coordinate frame to transform between frames. The resulting terrain map is sparse and nonuniform and creates two problems. First, the interpolation required to fill in the map is prone to error. Second, the algorithms required to access geometric information at arbitrary map locations are complicated.

Manuscript received November 30, 1990; revised January 7, 1991. This work was supported by NASA under Grant NAGW-1175.

I. S. Kweon was with the Vision and Autonomous Systems Center, the Robotics Institute, Carnegie Mellon University, Pittsburgh, PA. He is now with Toshiba Research and Development Center, Toshiba Corporation, Kawasaki, Japan.

T. Kanade is with the Robotics Institute, Carnegie Mellon University, Pittsburgh, PA 15213.

IEEE Log Number 9102690.

In rugged terrain, geometric sensors are frequently unable to see all the terrain within their field of view due to shadowing (occluding) by terrain geometry, that is, some terrain areas are not visible to the sensor. Identifying and representing invisible areas in the terrain model is critical for planning. With traditional Cartesian-based approaches, it is difficult to discern occluded areas due to sparseness in the terrain representation. Furthermore, determining upper bounds on terrain elevations within invisible areas is not possible. Finally, every sensor has inherent error in its data. the conversion of error in the sensor frame to a uncertainty model in the world reference frame is a nontrivial problem.

Shadowing, data sparseness, and error, especially at large distances from the sensor, limit the completeness of the data in a single image; the geometry of large areas of terrain is unknown due to shadowing. A forward-looking image alone may be insufficient for planning and navigation. Robots operating in rough terrain may require knowledge of terrain that has been observed but is currently out of the sensor field of view such as terrain under and behind the robot. Therefore, the need is to accurately merge successive images. For merging images, an estimate of relative displacement is required. Existing approaches, which estimate the displacement from dead reckoning or feature-based matching, fail to accurately merge images. For example, dead reckoning in rugged terrain has been shown to have 5% error in distance measure. Alternately, feature-based matching that finds correspondence between low-level features, such as points, lines, or corners, is not suitable for amorphous rugged terrain. More accurate displacement between images must be known before merging.

Even with good merging, some displacement error for each successive image merge is foregone. After many images, the resulting composite terrain map may accumulate a large error with respect to a world reference coordinate frame. The error in estimated robot position in the world coordinate frame may also be large. To solve this problem, global terrain geometric data independent of robot sensed local data is required. Error can be removed from the composite terrain map (and robot world position) by matching the composite map to the global elevation data.

In summary, the problems of modeling rugged terrain accurately include the following:

- How to model the immediate environment from a single range image
- how to estimate the incremental displacement between images before merging
- how to remove error accumulated in a composite terrain map by using global terrain data.

The goal of this work is to develop a 3-D vision system for building accurate and reliable representations of rugged terrain that are suitable for robot navigation.

II. Building a Grid Representation From a Range Image

In developing any 3-D vision system, selecting good representations will determine the robustness and performance of the system. The choice of representation should be based on the characteristics of the environment and the specific applications. For the exploration of Mars, the representation should be suitable for representing rugged terrain and for extracting other information such as obstacles for navigation or topographic features for sampling [1]. We then need to select a sensor suitable for building the representation.

In this section, we first present our representations (elevation maps) suitable for modeling 3-D rugged terrain. We then describe an active

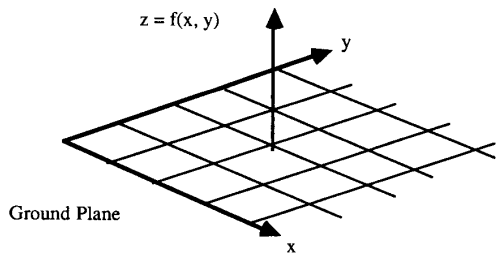


Fig. 1. Structure of an elevation map.

ranging sensor that creates range images. Finally, this section presents the locus method for building an elevation map from a range image.

A. Grid-Based Representation: Elevation Maps

Our representation for rugged terrain is a grid-based representation: an elevation map. Elevation maps represent the geometry of the environment by the elevation at grid points on a reference plane. The elevation is a vertical distance above or below the given reference surface discretized into a regularly spaced grid. an elevation map is then represented as $z = f(x, y)$ and is shown in Fig. 1.

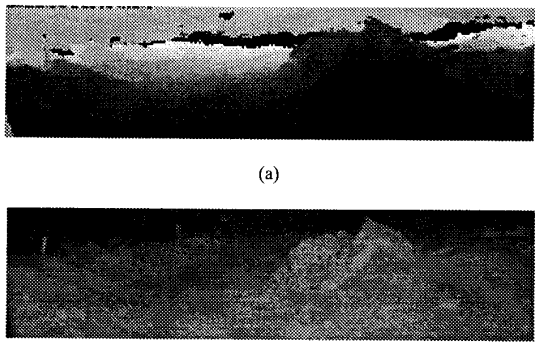
B. Active Ranging Sensor

Range or *depth* images can be created using active or passive range sensing methods. Stereo vision is a typical example of passive sensing methods. It finds the corresponding points from two images and computes the range by a triangulation. The fundamental problems in stereo are the difficulty and the computational costs of finding the *corresponding points* in the two images [6], [16]. Active sensing methods are preferred for two main reasons: First, they directly provide range data without the triangulation computation; second, active methods are largely insensitive to outside illumination conditions, simplifying considerably the image analysis problem. This is especially important for images of outdoor scenes in which illumination cannot be controlled or predicted. We have used a time-of-flight laser scanner manufactured by the Environmental Research Institute of Michigan (ERIM).

The ERIM laser scanner transmits a laser beam in the near-infrared region (810 nm) generated by a laser diode. The amplitude of the output of the laser diode is modulated by varying its drive current. It scans the beam across the field of view using a nodding mirror and a polygon mirror. The nodding mirror changes the elevation (tilts), and the polygon mirror changes the azimuth (pans) of the emitted signal.

The infrared light is reflected off the target surface, gathered by the receiver optics, and focused onto the detector, which is a silicon avalanche photodiode. It filters the optical signal to pass only the transmitted optical frequency and filters the electronic detector signal to pass only the amplitude modulating frequency. An electronic phase detector then measures the phase difference between the transmitted signal and the received signal, which is proportional to the transit time and, therefore, the range. Since relative phase differences can only be determined modulo 2π, the measured range to a point is really only determined to within a range ambiguity interval; for the ERIM, it is 64 ft. In addition to range images, the sensor also produces active reflectance images of the same format ($64 \times 256 \times 8$ b); the reflectance at each pixel encodes the energy of the reflected laser beam at each point.

Fig. 2 shows a pair of range and reflectance images of an outdoor scene. The range is coded in 8-b gray levels from zero to 64 ft. The reflectance at each pixel encodes the energy of the reflected laser beam in 8 b.



(a)

(b)

Fig. 2. ERIM images of outdoor terrain: (a) Range image; (b) reflectance image.

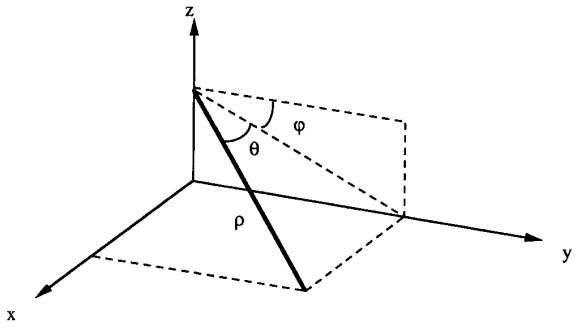


Fig. 3. Geometry of the ERIM range sensor.

The position of a point in the Cartesian coordinate system can be derived from the measured range and the direction of the beam at that point. Fig. 3 shows the range measurement (ρ) and the direction of the beam specified by the vertical (ϕ) and horizontal (θ) scanning angles. The two angles are derived from the row and column position in the range image (r, c) by the equations

$$\theta = \theta_0 + c \times \Delta\theta$$
$$\phi = \phi_r + r \times \Delta\phi \tag{1}$$

where θ_0 and ϕ_0 are the staring horizontal vertical angels, respectively. $\Delta\theta$ and $\Delta\phi$ are the angular step between two consecutive columns and rows, respectively.

From a simple trigonometry, the coordinates of a point measured by the ERIM range sensor are given by

$$x = \rho \sin\theta$$
$$y = \rho \cos\phi \cos\theta$$
$$z = \rho \sin\phi \cos\theta. \tag{2}$$

C. From Range Image to Elevation Map: The Locus Method

We could compute a Cartesian elevation map (x, y, z) by applying (2) to the measurements (ϕ, θ, ρ) in a range image; we call it the traditional approach. This approach has a few problems, as depicted in Fig. 4: First, this introduces nonuniform samples in Cartesian space. Even though the range finder scans with equal-angle intervals, the Cartesian map is progressively more sparse at points further from the sensor.

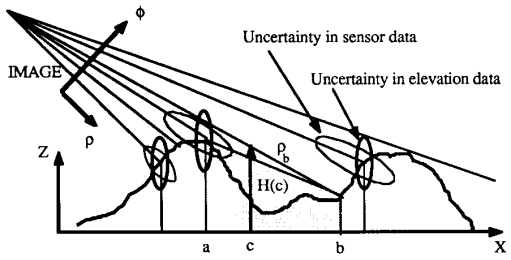


Fig. 4. Nonuniform samples, range shadows, and uncertainties in Cartesian space.

Second, objects in the environment may cast range shadows (e.g., the shaded areas in Fig. 4) created by rugged terrain. Without explicit information about the shadow areas, the surface would be smoothly interpolated, possibly incorrectly, and once converted into Cartesian space, it may be difficult to detect shadow areas.

Third, the uncertainty of range measurement is distributed across a region in the elevation map (e.g., the ellipse A in Fig. 4 covers many grid points in Cartesian space). Suppose that we want to compute an elevation value at a grid point b in Fig. 4. In this particular example, the elevation b may be determined by either one of two noisy range measurements whose uncertainties are shown by the ellipse A and B, respectively. In other words, uncertainty in the elevation is dependent not on a single range measurement but on many neighboring sensor measurements. To identify the uncertainty of the elevation value, we transform the uncertainty on a sensor measurement so that it is oriented along the z axis. This conversion is nontrivial for the traditional approach since the range uncertainty is distributed across a region in the elevation map.

The locus method uses a model of the sensor and works in image space, solving these problems associated with the traditional method. In the locus method, the elevation z at a point (x, y) on the reference plane is found by computing the intersection of the terrain with a vertical line at (x, y). At a grid point (x, y) on the horizontal xy plane, assume that we have nothing in the world but a vertical line (e.g., a vertical line $H(X)$ in the top figure of Fig. 5).

By computing the distance from the sensor to the scene points on this vertical line, we can create a locus of this vertical line in image space, which is labeled as *Locus,* in the bottom figure of Fig. 5. We derive the equation of the locus as a function of ϕ from (2), assuming x and y constant:

$$\rho = \rho_l(\phi) = \sqrt{\frac{y^2}{\cos^2\phi} + x^2} = \sqrt{\frac{y^2 + x^2\cos^2\phi}{\cos\phi}}$$

$$\theta = \theta_l(\phi) = \arctan\frac{x\cos\phi}{y}. \tag{3}$$

Now, we take a range image of the real scene without the hypothesized vertical line. The range image can be viewed as a surface $D = I(\phi, \theta)$ in ϕ, θ, D space. The basic idea of the locus algorithm is to compute this intersection point in image space rather than Cartesian space. A curve, labeled as *DepthProfile* in the bottom of Fig. 5, represents 1-D range image of the terrain. We find the intersection between the locus curve and the surface D (the intersection point, labeled *Intersection,* in Fig. 5). With a corresponding depth value of this intersection point, we can compute the elevation z at (x, y) by (2).

1) Algorithm: To obtain a basic locus curve, we project a vertical line onto the range image. The vertical line can be represented by

$$l = (u, v) = \left([u_x, u_y, u_z]^t, [v_x, v_y, v_z]^t\right) \tag{4}$$

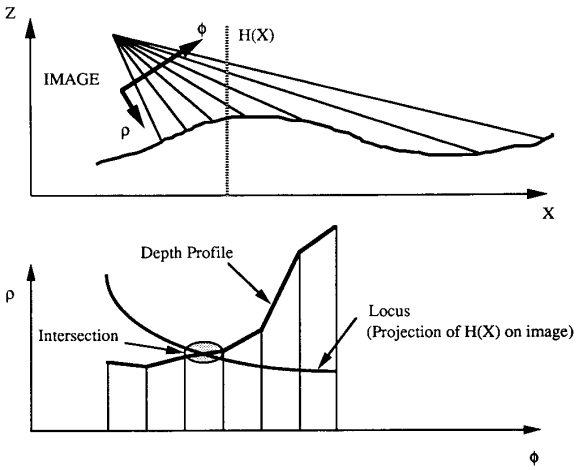


Fig. 5. Top figure shows imaging geometry: $\phi - \rho$ image plane, $X - Z$ Cartesian coordinate, and the terrain. $H(X)$ indicates a hypothesized line on the terrain. The bottom figure shows the projection of the terrain and the hypothesized line onto the image plane, noted as *DepthProfile* and *Locus.*

where u and v represent a point and a unit directional vector, respectively. Such a line is parameterized in λ by the relation $p = u + \lambda v$ if p is a point on the line. The basic locus can be obtained by projecting this vertical line onto the range image plane with (2). The basic locus method uses the intersection point between this basic locus and the surface observed by the sensor.

We now state the locus method algorithm for computing the elevation z at a grid point (x, y):

1) Compute the locus $\rho_l(\phi_j)$ by (3).
2) Compute the corresponding $\theta_l(\phi_j)$ by (3).
3) Obtain a sample data at (ϕ_j, θ_l), $\rho_m(\phi_j)$, from the range image.
4) Compute the difference between the locus and the sample of data, $\Delta(\phi_j) = \rho_l(\phi_j) - \rho_m(\phi_j)$.
5) Find the two scanlines of the range image ϕ_1 and ϕ_2 between which the intersection is located when $\text{sgn}\ \Delta(\phi_1) \neq \text{sgn}\ \Delta(\phi_2)$.
6) Apply a binary search between ϕ_1 and ϕ_2, and find the intersection point ϕ_n.
7) Compute ρ and θ by (3) and then compute an elevation value by (2).

Repeating the above procedures for vertical lines at every desired (x, y) point yields a dense elevation map with a desired resolution.

In summary, the locus method computes an elevation value from the intersection between the depth profile of image i and the locus of a line $l = (u, v)$. Let $f_i(u, v)$ represent a 3-D coordinate point computed from this intersection point:

$$f_i(u, v) = \begin{bmatrix} x \\ y \\ z \end{bmatrix} \tag{5}$$

where (x, y) is the horizontal position of the line, specified by u, and z is the estimate of an elevation at (x, y) from the locus method.

2) Range Shadows: We observe that a range shadow occurs along an occluding edge in the image. In Fig. 4, for example, a grid point c in the map is in a shadow area if its locus intersects the image at a pixel that lies on an occluding edge. We implement this idea by first detecting edges in the range image by using the GNC algorithm [2]. Then, in applying the locus algorithm, when the locus of a given

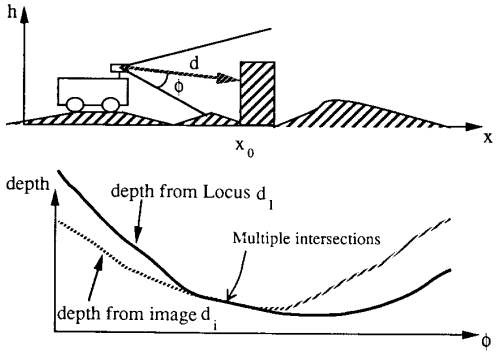


Fig. 6. Detection of multiple heights in the locus method.

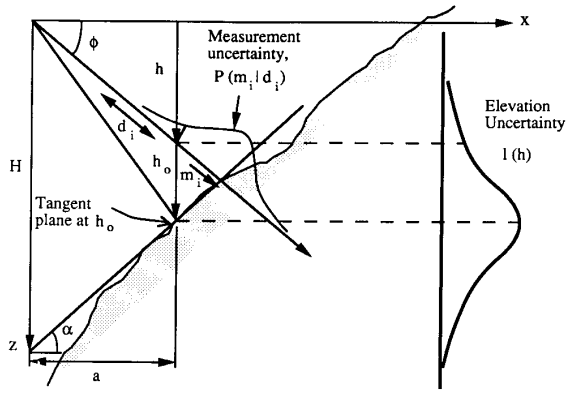


Fig. 7. Computing the elevation uncertainty from the sensor error model.

location intersects the image at an edge pixel, we mark that location as lying in a range shadow.

In addition to detecting shadow areas, the locus method can compute the upper bound of elevation values in the shadow area. In Fig. 4, for example, an elevation value at a point c cannot be greater than $H(c)$ because we have a range measurement ρ_b from the sensor.

3) Multiple Heights: One drawback of the elevation map is that we cannot represent multiple heights at the same grid point. Such representation is necessary to deal with vertical planes or overhung objects. We solve this problem by simply computing the maximum and minimum heights for the point.

Traditional approaches detect multiple heights by counting the number of points in the range image that are mapped to the same grid point on the elevation map. The locus method reports multiple heights when there are multiple intersections between the depth profile of the scene and the locus of a vertical line at a grid point (see Fig. 6). For a perfectly vertical plane, the locus coincides with the depth profile of the scene. Therefore, we are able to extract any vertical planes by just observing the multiple intersection points. This capability is very useful in segmenting range images for indoor scenes with vertical walls or outdoors scenes with buildings.

4) Uncertainty: The range sensor returns values with a limited resolution due to digitization. Those values are also corrupted by random measurement noise. We have developed a probabilistic model of the uncertainty on the sensor measurements. The uncertainty of range measurements is modeled to normally distribute with standard deviation proportional to the square of measured range [13] and to be oriented along the direction of measurement (see Fig. 4). To calculate the uncertainty on the elevation value at each grid point (x, y), we must transform the uncertainty of a range measurement so that the uncertainty along the z axis [8] is represented.

We use the nature of the locus method for computing the elevation uncertainty from the uncertainties of range measurements. Fig. 7 shows the principle of the uncertainty computation by considering a locus curve that corresponds to a line in space and the depth profile from the range image in the neighborhood of the intersection point. Consider a hypothesized surface point $p_i = (x, y, h_i(x, y))$ along the vertical line. The elevation h_i corresponds to a measurement direction $o_i(h)$ and a measured range $m_i(h)$. Let $d_i(h)$ be the distance between the origin and the surface point p_i. When we are thinking of the sensor error model $p(m|d)$ as a function of d, it is called the likelihood function and given by [14]

$$l(d|m) = p(m|d). \tag{6}$$

More specifically, the likelihood that this measurement $m_i(h)$ resulted from the surface point p_i can be described by our sensor error model $p(m|d)$ as follows [21]:

$$l(h) = l(d_i|m_i) = p(m_i|d_i) = \frac{1}{\sqrt{2\pi}\,\sigma(m_i(h))} e^{-\frac{(d_i(h)-m_i(h))^2}{2\sigma(m_i(h))^2}} \tag{7}$$

where $\sigma(m_i(h))$ is the variance of the measurement at the range $m_i(h)$.

In order to determine the shape of $l(h)$, we approximate $l(h)$ around h_o by replacing the surface by its tangent plane at h_o. Let α be the slope of the plane, and let H be the elevation of the intersection of the plane with the z axis. Assuming $\sigma(d) \approx Kd^2$ and using similar triangles, we compute $\sigma(m_i(h))$ as

$$\sigma(m_i(h)) = Km_i^2 = K\frac{H^2(a^2+h^2)}{(a\tan\alpha + h)^2} \tag{8}$$

where a is the distance between the line and the origin in the $x-y$ plane. Using (8), the exponent of (7) becomes

$$\frac{(d_i(h)-m_i(h))^2}{2\sigma(m_i(h))^2} = \frac{(h-h_o)^2(a\tan\alpha + h)^2}{2K^2H^4(a^2+h^2)}. \tag{9}$$

By assuming that h is close to h_o, that is $h = h_o + \epsilon$ with $\epsilon \ll h_o$, and by using the fact that $H = h_o + a\ \tan\ \alpha$, we have the approximations

$$\sigma(m_i(h)) \approx K(a^2 + h_o^2) \tag{10}$$

$$\frac{(d_i(h)-m_i(h))^2}{2\sigma(m_i(h))^2} \approx \frac{(h-h_o)^2}{2K^2H^2(a^2+h_o^2)}. \tag{11}$$

In the neighborhood of h_o, (11) shows that $(d_i(h) - m_i(h))^2/2\sigma(m_i(h))^2$ is quadratic in $h - h_o$ and that $\sigma(m_i(h))$ is constant. Therefore, $l(h)$ can be approximated by a Gaussian distribution of variance:

$$\sigma_h^2 = K^2H^2(a^2 + h_o^2) = K^2H^2d_o^2. \tag{12}$$

Equation (12) provides us with a first-order model of the uncertainty of h derived by the locus algorithm taking into account the uncertainty of the sensor measurements.

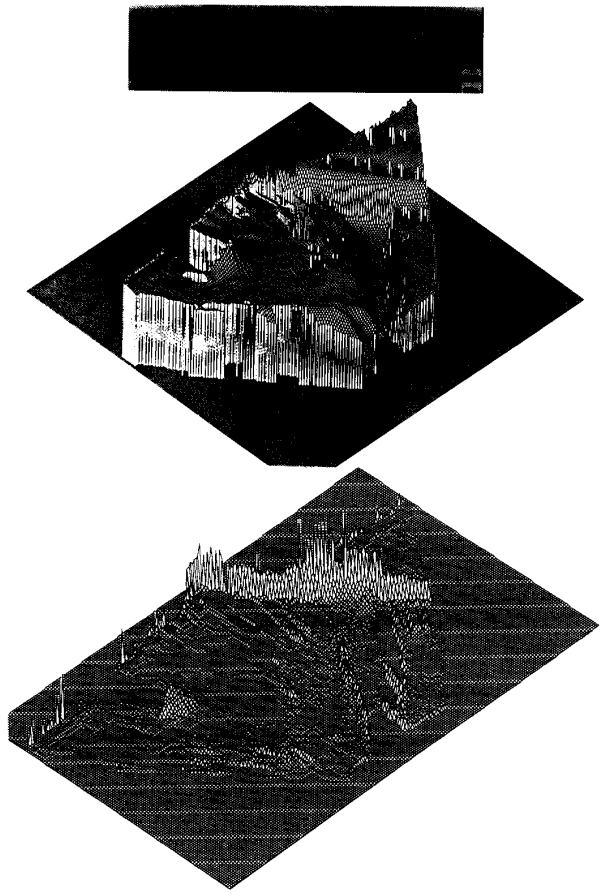


Fig. 8. Result of the locus method.

5) Results on ERIM Range Images: We applied the locus method to a real range image of rugged terrain. Fig. 8 shows an original range image, the resulting elevation map, and the uncertainty map computed by the locus method. The resolution of the map is 10 cm.

In summary, the locus method computes dense and uniform elevation maps at a desired resolution and easily detects shadow areas by working in image space rather than Cartesian space. The locus method also provides a mechanism necessary to transform the uncertainty of range measurements into the uncertainty of elevation data. The locus method is also applicable to other sensors, such as a dense stereo map and a light stripe range finder.

D. The Generalized Locus Method

We can generalize the locus method from the case of a vertical line to the case of a general line in space. This generalization allows us to use any reference plane instead of being restricted to the horizontal plane in building an elevation map. This is important when, for example, the sensor's (x, y) plane is not orthogonal to the gravity vector.

In Fig. 9, a line $H(X)$ is a vertical line with respect to the coordinate 1. In order to compute an elevation value at a grid point X, we apply the basic locus method to the image captured at vehicle position 1. Suppose that a new range image is captured at the location 2, and we want to estimate an elevation at the same map point. We could project the line $H(X)$ onto the new range image. However, the

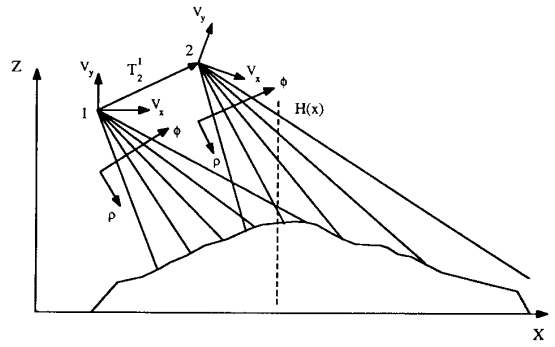


Fig. 9 Generalized locus method for map matching. The method computes an elevation at the same grid point by projecting the transformed line onto range image 2. A coordinate transformation of $H(X)$ by T_2^1 gives the transformed line.

line $H(X)$ is not a vertical line with respect to the coordinate 2, At the new sensor location, the line is represented by a transformed line

$$(u', v') = (Ru + t, Rv) \tag{13}$$

where R and t denote the rotation and translation parts of the transform between two viewing positions (e.g., T_2^1 in Fig. 9).

A generalized locus can be obtained by projecting this general line onto the range image. The generalized locus method computes an elevation by the intersection between the generalized locus and the depth profile.

A general line is still a curve in image space that can be parameterized in ϕ. The equation of the curve becomes

$$\begin{aligned} D_l(\phi) &= \sqrt{(v_x\lambda(\phi) + u_x)^2 + (v_y\lambda(\phi) + u_y)^2 + (v_z\lambda(\phi) + u_z)^2} \\ \theta_l(\phi) &= \arcsin \frac{v_x\lambda(\phi) + u_x}{D} \\ \lambda(\phi) &= \frac{u_y \tan\phi - u_z}{v_z - v_y \tan\phi}. \end{aligned} \tag{14}$$

We can then compute the intersection between the curve and the image surface by using the same locus algorithm as before except that we have to use (14) instead of (3).

The representation of the line by (4) is not optimal since it uses six parameters, whereas only four parameters are needed to represent a line in space. This can be troublesome if we want to compute the Jacobian of the intersection point with respect to the parameters of the line. A better alternative [9] is to represent the line by its slopes in x and y and by its intersection with the plane $z = 0$ (see [20] for a complete survey of 3-D line representations). The equation of the line then becomes

$$\begin{aligned} x &= az + p \\ y &= bz + q \end{aligned} \tag{15}$$

We can still use (14) to compute the locus because we can switch between the (a, b, p, q) and (u, v) representations by using the equations

$$\begin{aligned} v &= \begin{bmatrix} a \\ b \\ 1 \end{bmatrix}, & u &= \begin{bmatrix} p \\ q \\ 0 \end{bmatrix} \\ a &= \frac{u_x}{u_z}, & p &= -\frac{v_x}{u_z}v_z \\ b &= \frac{u_y}{u_z}, & q &= -\frac{v_y}{u_z}v_z. \end{aligned} \tag{16}$$

In the subsequent sections, we denote by $f(a, b, p, q)$ the function that maps a line in space to the intersection point with the range image.

In summary, the terrain representation with the locus method is

- Simple, compact, and easy to use
- uniform for other sensor data
- able to represent more rugged terrain by simply increasing the number of planar patches
- able to represent occluded areas explicitly
- able to compute elevation bounds for occluded areas
- able to represent the uncertainties of height values
- the basis for computing other information when needed.

III. Local Terrain Matching

Merging maps into a composite map can a) increase the resolution of those portions of the terrain map that were originally measured at a distance from the vehicle and b) add information about previously occluded areas. To build such a composite map from successive images, we need to estimate the vehicle position by matching the current terrain data with previous terrain data.

Recently there has been increasing interest in terrain matching for outdoor mobile robots [19], [5]. Two terrain matching algorithms (feature-based and pixel-based (iconic)) have been studied. Feature-based methods that match low-level features, such as points, lines, or corners, provides a good initial estimate of the vehicle position for iconic matching but are not suitable for unstructured natural scenes [5]. Iconic matching has been successfully applied to incremental depth estimation [15] and map matching [22]. However, it requires a good initial estimate of the vehicle position to reduce computational complexity.

To circumvent these problems, we use feature and iconic matching together. Our feature matching computes an estimate of the vehicle motion by extracting 3-D features from the elevation maps and identifying correspondences between them. This estimate is used as an initial estimate for our iconci matching. The iconic matching algorithm is based on the *generalized locus* methods and iteratively adjusts the vehicle displacement to minimize the accumulated error in the composite terrain map.

A. Feature Matching

Given two elevation maps, we compute an estimate T_F of the transformation between the two using the correspondences between 3-D features, such as high curvature points and lines extracted from the maps.

1) Feature Extraction: We use elevation map to calculate the curvatures of the surface as follows [3]: 1) Smooth the elevation map, 2) compute the first and second directional derivatives of the surface, and 3) compute the two principal curvatures by solving the first and second fundamental form. We use thresholding to find points of high surface curvature F_i^1 and F_j^2 from the elevation map 1 and 2, respectively. Fig. 10 shows the high curvature points extracted from an elevation map. The two images correspond to the two principal curvatures. 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 (e.g., see Fig. 11).

2) Feature Correspondence: Given two sets of features $\{F_i^1\}$ and $\{F_j^2\}$ extracted from elevation maps 1 and 2, we compute the displacement between the two maps by finding corresponding features from each map. The best correspondence determines the transformation T_F such that $F_{jk}^2 \approx T_F(F_{ik}^1)$.

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

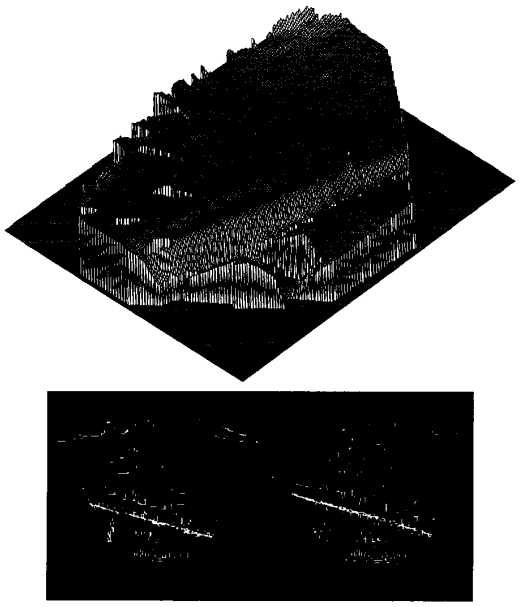
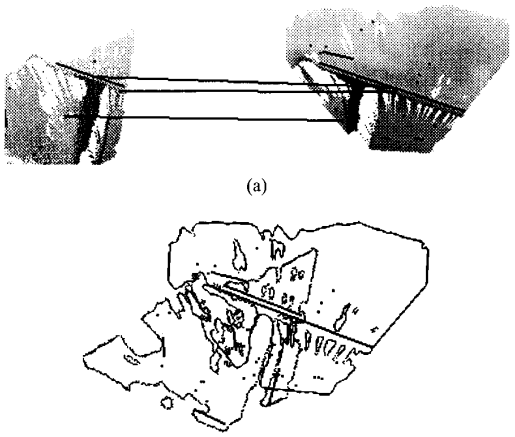


Fig. 10. Results of geometric terrain feature extraction. High curvature points.



(a)

(b)

Fig. 11. Results of feature matching.

S_k induces a rigid transformation T_{Fk}. We evaluate each candidate transformation T_{Fk} by computing the distance D between the features by

$$D = \sum_k d(F_{ik}^1, T_{Fk}(F_{ik}^2)) \tag{17}$$

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 that minimizes D, say S_k, is the final match between the two maps, and T_{Fk} is the best transformation.

Fig. 11 shows the result of the feature matching on a pair of elevation maps. Fig. 11(a) shows the correspondences between the point and line features in the two maps. Fig. 11(b) shows the

superposition of the contours and features of the two maps using the estimated displacement (the left map is transformed by T_{Fk} with respect to the right map).

B. Iconic Matching

The feature matching method described above computes a motion estimate using correspondences between features. However, in 3-D rugged terrain, determining correspondences may be difficult because of occlusions and amorphous terrain. To solve this correspondence problem, we have developed an iconic matching method based on the generalized locus method.

Let $f_i(u, v)$ be the function that represents the locus method (i.e., computes an elevation value from the intersection between depth profile of image 1 and the locus of a line $l = (u, v)$). The vector $f_i(u, v)$ denotes the coordinates of a particular point

$$f_i(u, v) = \begin{bmatrix} x \\ y \\ z \end{bmatrix} \tag{18}$$

where (x, y) is the horizontal position of the line, specified by u, and z is the estimate of an elevation at (x, y) from the locus method.

The function $f_1(u, v)$, for example, computes an elevation from the intersection of a line passing through a grid point u with image 1. The function $f_2(u', v')$ computes the elevation in map 2 of the same grid location for which the elevation had been computed in map 1. The transformed line (u', v'), can intersect anywhere in image 2 and is not a vertical line. Since the generalized locus algorithm computes $f_2(u', v')$ by finding this intersection point, we can directly compare the two elevations without determining correspondences.

To compare two elevation estimates, we transform f_2 to the coordinate of image 1 by a transform T_2^1. We can represent the transformed function as follows:

$$g(u, v, T_2^1) = T_2^{1^{-1}}(f_2(u', v')) = R'(f_2(u', v')) + t' \tag{19}$$

where

$$T_2^{1^{-1}} = (R', t') = (R^{-1}, -R^{-1}t), \quad \text{and}$$
$$(u', v') = (Ru + t, Rv).$$

1) Error Function: Given the starting estimate T_F from either a feature matching or an on-board position sensor and a pair of range images, the iconic matching algorithm computes the transformation T_2^1 that minimizes a cost function E by iterative gradient descent [7]:

$$E = \sum \| f_1(u, v) - g(u, v, T_2^1) \|^2 \tag{20}$$

where the summation is taken over all grid points where both f_1 and g are defined. The error function weighted by uncertainty is discussed in [10].

2) Gradient Descent: The error E reaches a minimum when $\frac{\partial E}{\partial \eta} = 0$, where $\eta = [\alpha, \beta, \gamma, t_x, t_y, t_z]$ represent the transformation parameters of which the first three are the rotation angles, and the last three are the translation vector. Assuming a reasonably accurate initial estimate of η by T_F, the minimum error can be achieved by an iterative gradient descent of the form

$$\eta^{i+1} = \eta^i + k \frac{\partial E}{\partial \eta}(\eta^i) \tag{21}$$

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

At each iteration, the algorithm a) computes $g(u', v')$ by applying the updated transformation to the measurements in range image 2, b) computes the error between the first and second measurements by (20), c) computes the partial derivative of the error function with respect to each of the transformation parameters, and d) updates the transformation parameters by (21).

Iteration continues until either the variation of error ΔE is small enough (convergence) or E itself is small enough.

3) Derivative of the Error Function: We compute the derivative of E from (20):

$$\frac{\partial E}{\partial \eta} = -2 \sum (f_1(u, v) - g(u, v, T_2^1)) \frac{\partial g}{\partial \eta}(u, v, T_2^1). \tag{22}$$

From (19), the derivative of g is

$$\frac{\partial g}{\partial \eta}(u, v, T_2^1) = R' \frac{\partial f_2}{\partial \eta}(u', v') + \frac{\partial R'}{\partial \eta} f_2(u', v') + \frac{\partial t'}{\partial \eta}. \tag{23}$$

We can compute analytically the derivatives appearing in the last two terms in (23). We can also compute the derivative of $f_2(u', v')$ with respect to η by the chain rule.

4) Problem with the Iconic Matching: The iconic matching method is computationally very expensive. To improve the computational efficiency, we employ coarse-to-fine-matching. The motion estimate from a coarser resolution is used to compute a more accurate motion for the next finer level of resolution.

We use three levels of resolution (coarse, medium, and fine) in which the resolution is halved successively. We first build a fine-level representation and simply subsample the representation to obtain representations at the coarser levels assuming that the coarse level grids are a subset of the fine-level grids. In the locus method, the implementation of building this multiresolution representation is almost trivial since the locus method can build a representation at any arbitrary resolution.

There are many sophisticated strategies to coordinate interactions between coarse and fine levels [23]. We use a simple coarse-to-fine algorithm: The motion estimate from a coarser resolution is used as the initial estimate to compute a more accurate motion for the next finer level of resolution.

IV. Building a Composite Terrain Map

We now present a method for building a composite terrain map from a sequence of range images by combining the terrain matching method and the locus method that we described in the previous sections. The locus method constructs an elevation map from a single range image and the terrain matching algorithm computes the transformation between two images. The terrain matching algorithm consists of the two steps: the initial estimate of the transformation by either feature matching or inertial navigation system (INS) data and the refinement of the initial estimate by the iconic matching method.

We first extract the relative motion between two successive images using the INS data, which is represented with respect to an absolute reference frame. Large error is avoided because there is no large INS drift between successive images, even though the INS data does drift over long distance. However, an error in the motion estimate for any particular pair of successive images affects all the subsequent mapping results because the error in the resulting composite terrain map is propagated through this erroneous motion estimate.

The detailed algorithm proceeds in the following steps (Fig. 12 shows the overall structure of the algorithm):

1) Construct an elevation map from image $k - 1$ using the locus method.
2) Compute a transform T_{k-1}^k between images $k - 1$ and k by the iconic matching method using the initial estimate of the transform from INS.

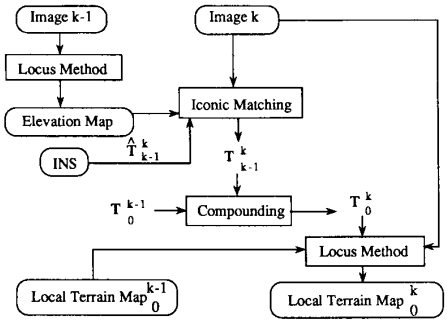


Fig. 12. Terrain mapping system flowchart.

3) Compute a global transform T_0^k with respect to a reference frame by compounding transforms computed up to the image $k-1$.
4) Apply the generalized locus method with T_0^k to the current image k, producing an elevation map.
5) Combine this elevation map with the *a priori* (or previous) composite terrain by
   - adding nonoverlapping points to the composite map, or
   - for overlapping points, replacing the elevation by the maximum likelihood estimate

$$f_1(u,v) \leftarrow \frac{\sigma_2^2 f_1 + \sigma_1^2 f_2}{\sigma_1^2 + \sigma_2^2} \tag{24}$$

   where f_1 and f_2 indicate the old and new elevation, and σ_1 and σ_2 are the standard deviations of the uncertainty distributions on the two elevation estimates
   - updating the variance by

$$\sigma_1^2 = \frac{\sigma_1^2 \sigma_2^2}{\sigma_1^2 + \sigma_2^2}. \tag{25}$$

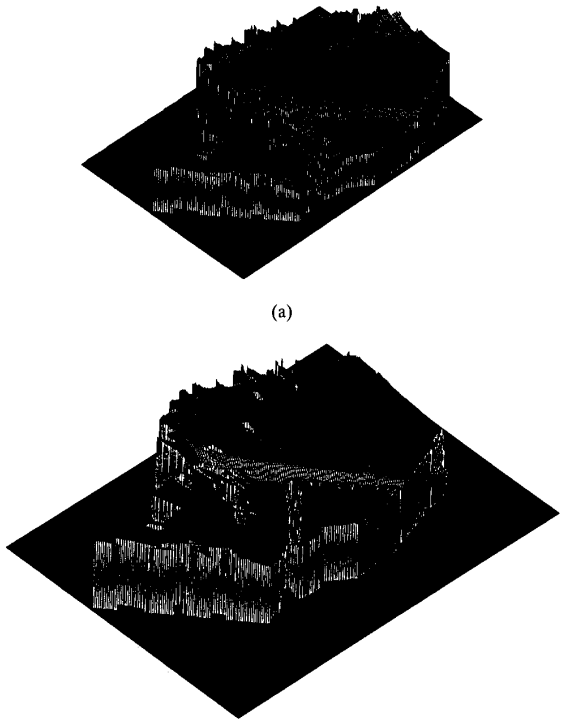
A. Experimental Results

In this section, we present two sets of experimental results of building a composite terrain map: one by feature and iconic matching and the other by INS data and iconic matching.

We have tested our terrain mapping system using sequences of both real and synthetic range images. For feature and iconic matching, we have used a set of real range images obtained at a construction site at Carnegie Mellon University (CMU). For INS and iconic matching, we have used two sets of real range images: one obtained at a slag pile area near CMU and the other taken at the Martin Marietta (MM) autonomous land vehicle (ALV) test site. All test sites were unstructured rugged outdoor terrain.

Using real range images poses many difficulties in verifying the results due to the lack of ground truth in terrain and vehicle motion data. Therefore, we use a terrain simulator to create synthetic range images and digital elevation maps (DEM's). These synthetic data are used for evaluating the accuracy of our terrain mapping system.

1) Terrain Mapping: Feature and Iconic Matching: In this section, we present the experimental results from our terrain matching method that combines the two complementary terrain matching methods: feature and iconic matching. As described in Section III-A, we compute the initial transforms between successive range images by a feature matching method. These transforms become initial estimates for the next iconic terrain matching step. The iconic matching method iteratively computes a least-squares solution of the transform starting with the given initial motion estimate.



(a)

(b)

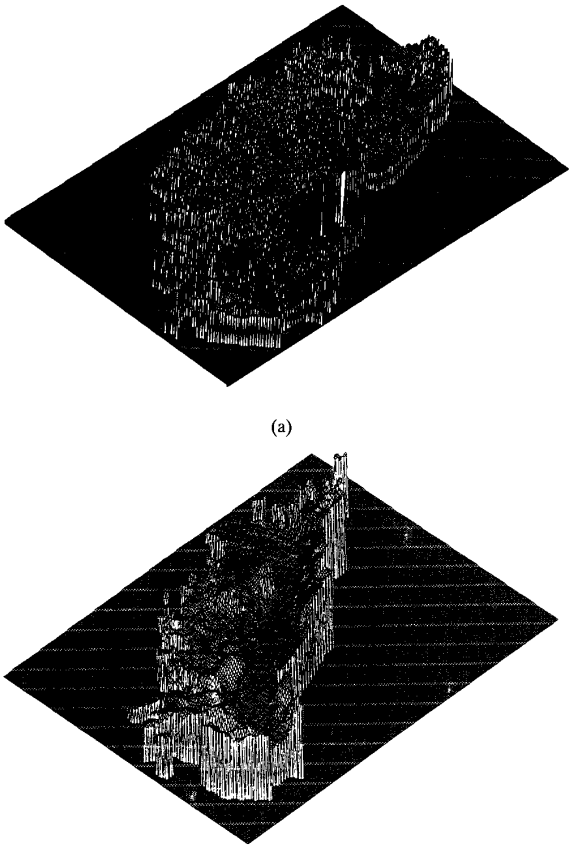
Fig. 13. Results for the construction test site.

The experiment to verify this terrain matching method has been done using a sequence of five range images from a construction site at CMU [11]. Fig. 13(a) shows the composite terrain map obtained by feature matching. We observe a large error especially along the boundaries between elevation maps. The iconic matching method creates a much more accurate terrain map, as shown in Fig. 13(b). From these experiments, we demonstrate that combining the two complementary matching methods into one terrain matching method is a feasible and robust approach for rugged terrain.

2) Terrain Mapping: INS Data and Iconic Matching: We have done three different sets of experiments to evaluate our terrain mapping system that combines INS data and iconic matching. First, we captured a short sequence of 40 range images and measured the robot's position at each image using a surveying instrument. To estimate the accuracy of our terrain mapping system, we compared the robot's positions computed by our iconic matching method with the measured ones. Second, we tested our system using a long sequence of 122 range images over the distance of 150 m. Lastly, we also tested our system with synthetic range images for which ground truth terrain data are available.

3) A Real World Experiment with Ground Truth Vehicle Positions: We have tested the accuracy of our system using a slag pile area near CMU. We captured 40 range images using CMU Navlab equipped with an INS system. For each image, we also measured the vehicle position by using a surveying instrument. The accuracy of this instrument is better than 1 cm at 100 m, and we used these measured positions as the ground truth vehicle positions.

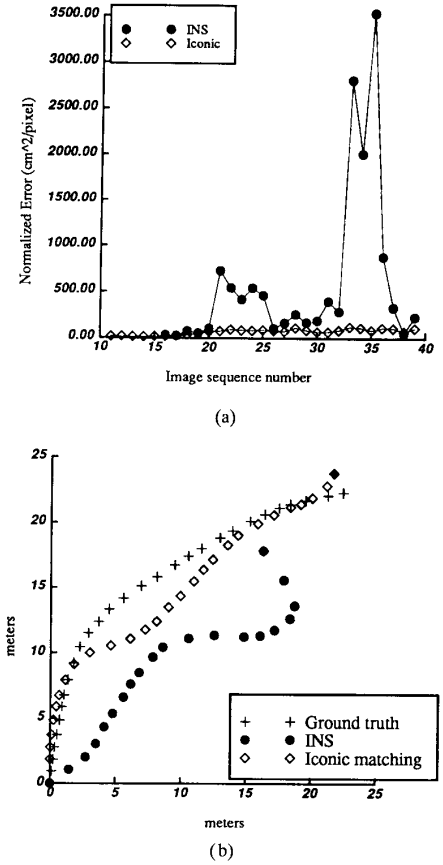
Fig. 14 shows the result of combining 40 real range images as a composite elevation map at 20 cm resolution. Fig. 14(a) shows the resulting composite map from the generalized locus method with INS data. This map shows a kind of random elevations that cannot



(a)

(b)

Fig. 14. Composite terrain maps for a slag heap area.



(a)

(b)

Fig. 15. Results for slag heap data.

be true for the test site. Using the iconic matching method in which INS data are used as an initial estimate of vehicle position, we were able to obtain a coherent terrain map that looks more reasonable in its appearance. Fig. 14(b) shows the resulting composite map by the iconic matching method.

Since we do not know the ground truth elevation data, we use the previous composite terrain map as ground truth to examine the error in the elevation data. We compute the mean-squared (MS) elevation error between the previous and current composite terrain map by

$$E = \frac{\sum_{i=1}^{N} (f_1 - f_2)^2}{N} \tag{26}$$

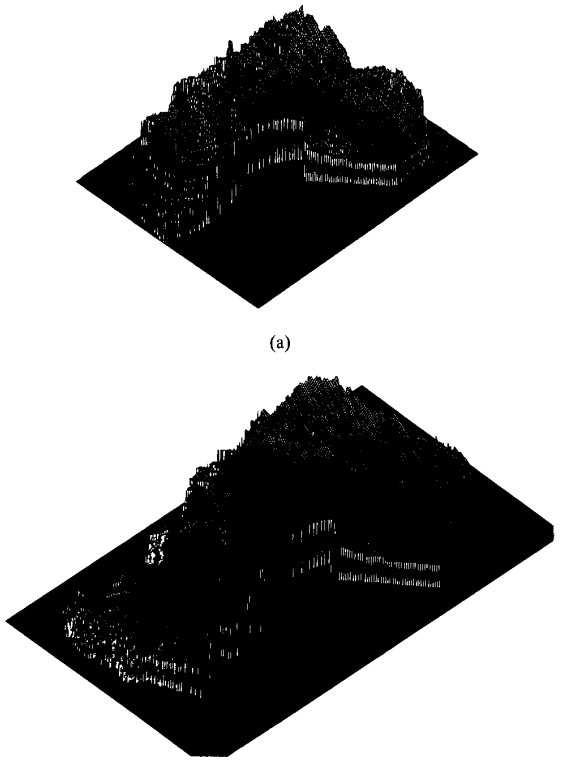
where f_1 and f_2 indicate the previous and current elevation data, and N is the number of data of the overlapped area.

Fig. 15(a) shows the MS elevation error E of the composite elevation map from our iconic matching method and INS data. Fig. 15(b) shows the resulting robot positions from a surveying instrument, the iconic matching method, and INS data. Using a surveying instrument, we measure the ground truth vehicle positions with the accuracy of 10 mm, and they are shown as plus symbols in the figure. Each diamond and black dot indicates an estimated robot position by the iconic matching method and INS data, respectively. We observe the large discrepancies between the estimated robot positions and the ground truth data starting from the 11th vehicle position. This is due to large error in the initial estimate of the robot positions by the INS data.[1]

The results of this experiment demonstrate the power of our terrain mapping system. Even though the INS data drift relative to the reference coordinate, the relative motion between two successive images is still sufficient for the iconic matching method to refine the motion estimate.

4) A Large-Scale Real World Experiment: To evaluate the performance of the terrain mapping system for a large number of images covering a much larger area, we tested our iconic matching algorithm by building a composite elevation map from a long sequence of real range images of the MM ALV test site. Over the travel distance of 150 m, we captured a sequence of 122 range images using MM ALV that also provided INS data. For comparison, we built a composite terrain map by merging range images based on only INS data. Fig. 16(a) shows the resulting terrain map. We do not observe large errors in the elevations up to the middle of the robot path. However, large discrepancies between the new map and the previously built map are observed around the end of the test run. This is due to the drift of INS data over the long distance of travel and the inaccuracy of INS in rugged terrain.

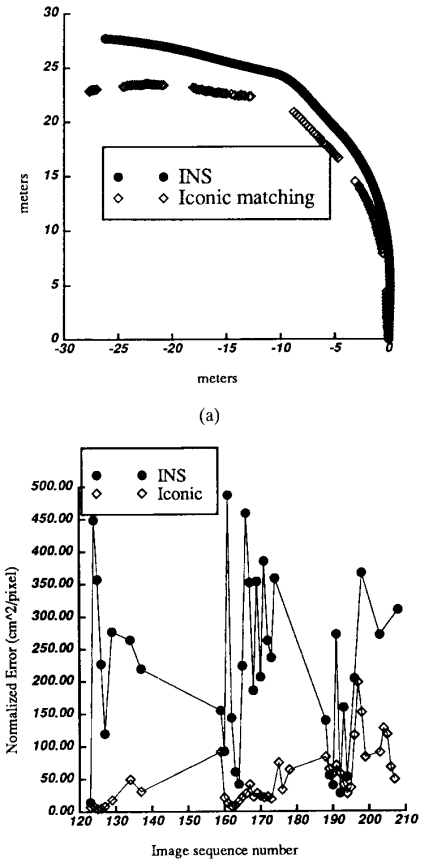
[1] For this particular experiment, we have used the vehicle positioning system (VPS), which is specifically designed for high speed navigation. Therefore, we have observed large fluctuations in both distance and angular displacements while acquiring range images by a stop-and-go method with a very slow vehicle speed.



(a)

(b)

Fig. 16. Composite terrain maps of combining 122 real range images from the ALV test site.



(a)

(b)

Fig. 17. Results for MM ALV test site.

Fig. 16(b) shows the resulting composite terrain map of combining 122 real range images by the iconic matching method. For these experiments, the INS data are used as the initial estimate of the unknown motion for the iconic matching method. This composite map shows a smooth terrain without any abrupt changes in elevations and paths of water flow, which qualitatively correspond to what we observed from the experimental site.

Fig. 17(a) shows the MS elevation error of the composite elevation map by our iconic matching method and INS data. The error is evaluated using the previous composite terrain map as ground truth using (26). Black dots and plus symbols indicate the error resulted from INS data and the iconic matching method, respectively. We observe that the iconic matching method remarkably reduced the error in the elevation values.

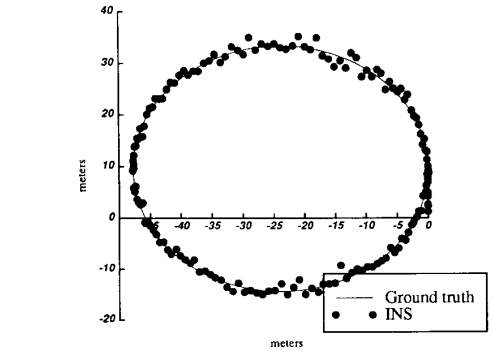
Fig. 17(b) shows the resulting robot positions from the iconic matching method and INS data. Each diamond and black dot indicates an estimated robot position by the iconic matching method and INS data, respectively. The discrepancies between the two robot paths are because estimates of initial vehicle roll are not available or range images are bad. For diamond symbols in Fig. 17(b), a few occasional jumps in the estimated vehicle positions are due to these problems. In those cases, we have used the estimate of motion from INS as our best estimate. Therefore, even a small angle error at those positions can cause a large error in the resulting composite terrain map since it is represented with respect to the starting vehicle position. This evaluation is verified in Section IV-A-2 by the results of experiments with synthetic data.

5) Large-Scale Synthetic World Experiments: A Terrain Simulator: Quantitative analysis of terrain mapping algorithms is a difficult process because of the difficulty in recovering ground truth data. To overcome this problem, we have developed a range sensor simulation tool and coupled it with a geometric modeling system [24]. Using this system, we can generate realistic 3-D terrain including additional features, such as rock distributions, and then generate a sequence of range images as we move the sensor through the environment. We can simulate the process of a vehicle moving through rugged terrain and compare the output of the sensor algorithms with the true terrain data used in the modeling system.

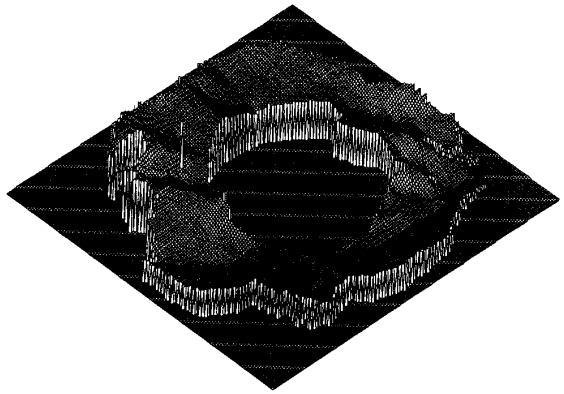
Terrain is modeled using a triangular mesh representation. Data for the mesh comes from a 5-m resolution DEM, which is scaled up to 50-cm resolution for the purposes of our experiments.

In order to efficiently simulate the range sensor, we have attempted to map much of its image formation process into available graphics hardware. Depth-cued images are produced using a hardware z buffer, and a Cartesian depth map is produced. This map is then transformed into spherical coordinates to simulate the ERIM's projection process. We oversample in the original perspective images to accomplish this transformation without loss of information.

With the terrain simulator, we created a sequence of 159 range images along a circular path with a radius of 50 m. It provides two sets of motion estimates between successive range images: the true motion estimate and the noise contaminated estimate. The error in motion is modeled as Gaussian noise and is added to the true motion. For this particular set of images, we contaminated the motion with

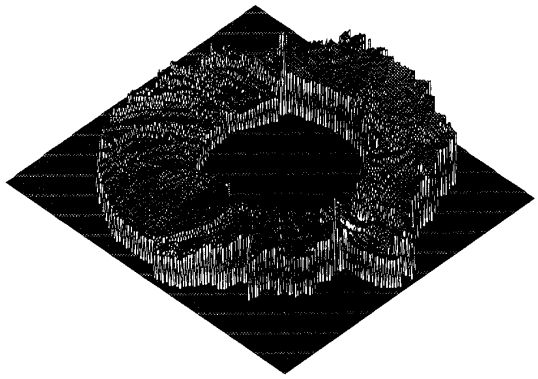


(a)

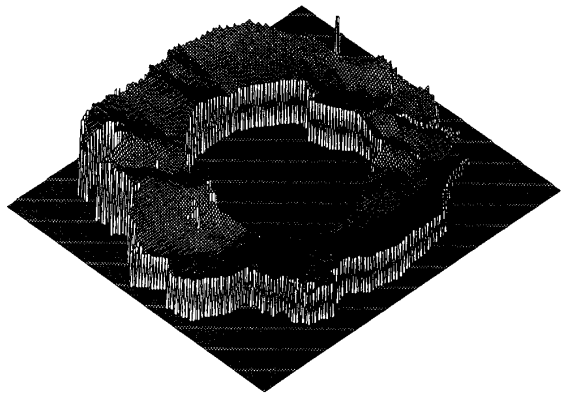


(b)

Fig. 18. Ground truth data for circular motion.



(a)



(b)

Fig. 19. Composite terrain maps computed from synthetic range images.

5% error[2] as follows:

$$M_{\text{noise}} = M + \frac{5M}{100}x \tag{27}$$

where M denotes the true distance or angular displacement along the x, y, and z axes and $x \sim N(0,1)$. Fig. 18(a) shows the vehicle trajectories for the experiments. The ground truth vehicle path is a perfect circle with the radius of 50 m. Over the distance of travel of 157 m, the simulator created a sequence of 159 range images. Black dots indicate the noise contaminated robot positions that simulate the INS data.

To evaluate the accuracy of the map produced by the iconic matching method, we first build the ground truth composite terrain map using the ground truth motion obtained from the terrain simulator.

The ground truth composite map is shown in Fig. 18(b). This true map is compared with a composite map obtained by the iconic matching method.

To determine the accuracy of the initial noisy motion estimate, we also built a composite terrain map using the noise-contaminated motion estimates. Fig. 19(a) shows the resulting terrain map. The composite terrain map shows false elevation data. Fig. 19(b) shows a composite terrain map computed using the iconic matching method. The iconic matching method used the noisy motion estimate from the simulator as an initial estimate. The resolution of the map is 20 cm.

Fig. 20(a) shows the MS elevation error of the composite elevation map by the iconic matching method and INS data.

[2]Inertial navigation systems (INS's) typically have 0.2% error in distance measure on hard surface and 5% error on natural terrain.

Fig. 20(b) shows the estimated vehicle positions on the ground plane (xy plane) for the circular trajectory. Dots and diamonds indicates the initial vehicle positions from the simulator and the estimated vehicle positions by the iconic matching method, respectively. The circle drawn with solid curve indicates the ground truth path. We can observe small errors between the true path and the computed path by the terrain matching method. For example, the error of the final position is 3 m, which is within 1% of the traveled distance. These errors are due to bad initial estimates of displacements for range images from 30 to 50 and from 110 to 125. We can observe large errors ($\sim$ 3 m) between the starting INS data and the true path for images from 30 to 50 and from 110 to 125.

From the results of experiments with real and synthetic range images, we conclude that the terrain mapping system using INS data and the iconic matching method creates an accurate composite terrain map from a long sequence of range images acquired over a large area of rugged terrain.

V. Global Terrain Matching

Global terrain matching estimates the vehicle position by matching the current terrain data from local observation with a DEM. Matching local observations with a DEM can also be used for improving the resolution and accuracy of the DEM. Over a long distance of robot motion, the error accumulates in a resulting composite terrain map if only local observations are used. However, a prior DEM can solve this error accumulation problem if we estimate the vehicle position in the DEM.

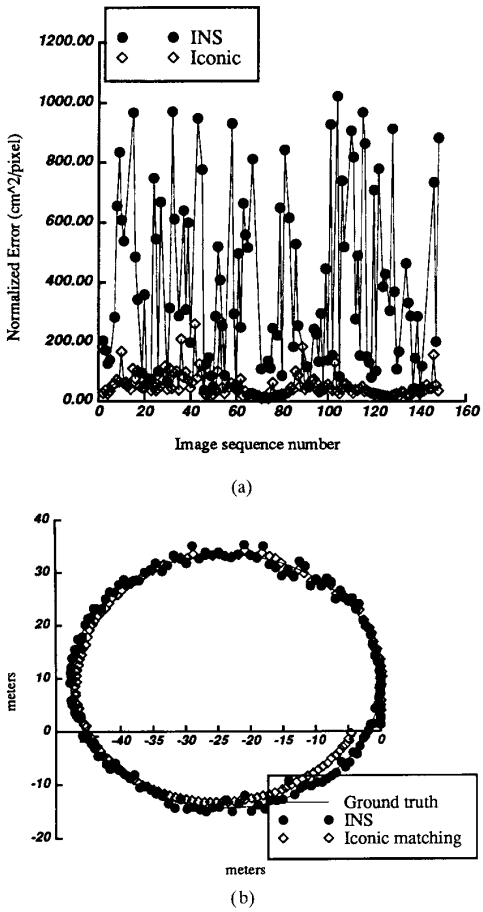


Fig. 20. Results from synthetic range images.

DEM's have been used for many years by geographers and cartographers for the measurement and analysis of the earth's surface. Recently, there has been considerable interest in using the DEM for 3-D terrain matching, that is, matching the local data obtained from stereo vision or laser range finder to a prior DEM, which we call the *local-global* matching. There are very few research works reported for the *local-global* matching method. Gennery [5] described an iconic matching method for height maps from a robot using stereo vision and orbital imagery. His terrain matching system assumes that the orientation of the vehicle is accurate and computes only the translation between height maps.

In this work, we present a terrain matching method to solve the *local-global* matching problem.

A. Digital Elevation Maps (DEM)

A DEM is a 2-D array of uniformly spaced terrain elevation measurements. In aerial photogrammetry, a DEM is constructed from aerial stereo photography either manually by using a plotting instrument or automatically by using stereo matching algorithms [18]. The elevations measured by most plotter systems have varying degrees of accuracy, depending on the precision of the system and the flying height, typically ranging from 1/1000 to 1/10 000 times the flying height [17].

We have used a DEM compiled by the U.S. Army Engineer Topographic Laboratories (USAETL) for an ALV test site in Colorado

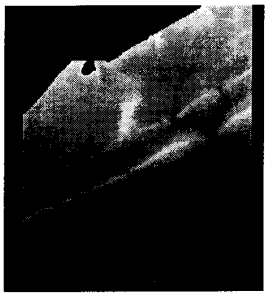


Fig. 21. DEM from the ALV test site in Colorado.

covering approximately 12 km^2 [4]. Elevations for terrain were collected with the resolution of 5 m using stereo photos at a scale of 1:12 000. Different areas were sampled at a variable range of sampling densities. For rugged areas, elevation data were collected at a wider spacing and were interpolated to a 5-m spacing. The accuracy of elevations ranges approximately from 1.5 to 5 m, depending on this spacing in the original elevation data. Edwards [4] explains that the error in elevations are observed at the region edges for the sparse posting density and at slopes for the dense posting density.

Fig. 21 shows the DEM where the darker pixels indicate lower elevations. We observe two large ridge lines across the terrain.

B. Terrain Matching Using a DEM

Matching the terrain data obtained from local observation with a DEM benefits us in two ways. First, we can compute the global position of the robot. Therefore, we can eliminate some of the systematic error, such as error due to bad calibration and the error accumulated while building a composite terrain map from local observations. Second, we can use local observations for improving the resolution and accuracy of the DEM.

In matching local observations with a DEM, we have two difficulties: 1) the accuracy of elevations (ranging 1.5 to 5 m) from the DEM is inferior to that of elevations maps ($\sim$ 10 cm) built from range images, and 2) the DEM has a relatively poor resolution (5 m) compared with the local observations.

The poor resolution of the DEM makes the global matching difficult because too few corresponding points for matching are available. To solve this problem, we use a sequence of range images instead of single image and estimate the transform between images by the iconic matching method.

In the following sections, we first describe how to use multiple images for our global terrain matching method. Then, we extend the generalized locus method into the box locus method to combine two different sensor data with very different error characteristics. Finally, we present the result of experiments on real range images and a DEM.

1) The Problem: Fig. 22 illustrates the global terrain matching problem. Let rectangles indicate the sequence of robot positions at which images are acquired. We want to estimate the current vehicle position, which is drawn as a rectangle k in Fig. 22, in a DEM. In other words, global terrain matching estimates the transform T_k^{dem} from the coordinate of the position k to the DEM coordinate dem.

A single image acquired at k can only cover a small area (the area shaded by deviant lines in Fig. 22) and provides only a few corresponding points between the DEM and the image. However, a sequence of images $k \cdots m$ covers a larger area (the shaded area in the figure) and can provide enough data points for the iconic matching. To use a sequence of images, we need to estimate all the transforms T_m^k occurring between viewing positions $k \cdots m$. This can

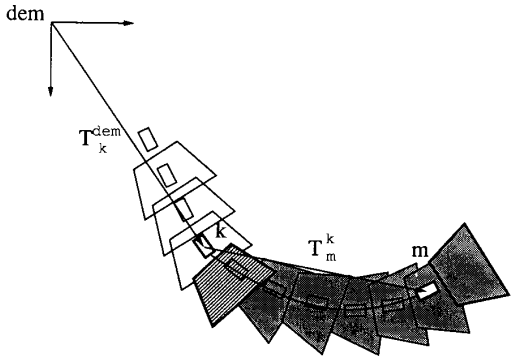


Fig. 22. Global terrain matching between range images and the DEM.

be done by using the iconic matching method described in Section III-B. The range images with the estimated transforms and a DEM are used to compute a transform T_k^{dem}.

2) The Box Locus Method: The locus algorithm computes an elevation from the intersection of the surface observed by the sensor and a vertical line passing through a grid point. For each grid point of the DEM, imagine a rectangle window whose size is the same as the resolution of the DEM. We compute four intersection points corresponding to four corner points of the window. Using all points inside four intersection points in the image space, we compute a mean elevation value.

In Fig. 23, all range measurements in shaded areas are used to compute an elevation value at a map point A. Shaded areas are bounded by two locus curves that are projections of the two vertical lines H_1 and H_2 onto the range image. Two vertical lines H_1 and H_2 are located at the distance of half the resolution from A. We obtain two locus curves in image space *Locus1* and *Locus2*, as shown in the left half of Fig. 23. Two intersections between the depth profile and locus curves defines a region, which is marked by shaded area in the figure. We use depth data inside this region to estimate the elevation at A. Since the local map and the DEM now have a comparable resolution, we can compare the data by directly applying the iconic matching method.

We still need an initial estimate of the transform between the coordinate of the robot and the DEM (T_k^{dem} in Fig. 22) to obtain the faster convergence with the iconic matching method. In this work, we assume that an initial estimation of the robot position relative to the DEM coordinate is given.

3) Algorithm for Global Terrain Matching: In the preceding sections, we described the two key concepts for matching the local terrain data from range images with the global terrain data. In this section, we present the detailed algorithm of global terrain matching.

We first derive the compound transforms necessary for the iconic matching method. For simplicity, we describe the global terrain matching algorithm for two successive range images denoted as image 1 and image 2. However, the same algorithm can be easily extended to the case of n range images. Let the subscript 0 refer to the origin of the DEM.

We want to estimate the transform $\boldsymbol{M}_{10} = (\boldsymbol{R}_{10}, \boldsymbol{t}_{10})$ from the image 1 coordinate to the DEM coordinate 0. Using the iconic matching method described in Section III-B, we can obtain an accurate estimate of the transform $\boldsymbol{M}_{21} = (\boldsymbol{R}_{21}, \boldsymbol{t}_{21})$ from the image 2 coordinate onto the image 1 coordinate. We can transform a point $\boldsymbol{p}_1$ in the coordinate 1 to a point $\boldsymbol{p}_2$ in the coordinate 2 by

$$\boldsymbol{R}_{21}\boldsymbol{p}_1 + \boldsymbol{t}_{21} = \boldsymbol{p}_2. \tag{28}$$

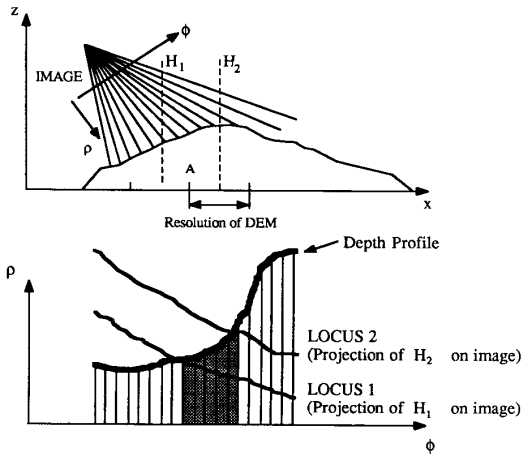


Fig. 23. Locus method for matching high-resolution local observations (range images) with the poor resolution DEM.

Now, we are ready to apply the locus method to global terrain matching. For the image 2, we need to compound the two transforms $\boldsymbol{M}_{10}$ and $\boldsymbol{M}_{21}$ for the iconic matching method as follows:

$$\boldsymbol{p}_2 = \boldsymbol{R}_{21}\boldsymbol{R}_{10}\boldsymbol{p}_0 + \boldsymbol{R}_{21}\boldsymbol{t}_{10} + \boldsymbol{t}_{21} \tag{29}$$

where $\boldsymbol{p}_2$ and $\boldsymbol{p}_0$ refer to 3-D coordinates of points in the coordinates 2 and 0, respectively. We abbreviate it

$$\begin{aligned}\boldsymbol{M}_{20} &= (\boldsymbol{R}_{20}, \boldsymbol{t}_{20}) \\ &= (\boldsymbol{R}_{21}\boldsymbol{R}_{10}, \boldsymbol{R}_{21}\boldsymbol{t}_{10} + \boldsymbol{t}_{21}).\end{aligned} \tag{30}$$

The inverse matrices of the compounded transforms are given by

$$\begin{aligned}\boldsymbol{M}_{20}^{-1} &= (\boldsymbol{R}'_{20}, \boldsymbol{t}'_{20}) \\ &= (\boldsymbol{R}_{10}^{-1}\boldsymbol{R}_{21}^{-1}, \boldsymbol{R}_{10}^{-1}\boldsymbol{t}_{10} + \boldsymbol{R}_{10}^{-1}\boldsymbol{R}_{21}^{-1}\boldsymbol{t}_{21}).\end{aligned} \tag{31}$$

We compute the derivative of E from (20) with respect to η:

$$\frac{\partial E}{\partial \eta} = -2\sum (f_1(u,v) - g(u,v,T_I))\frac{\partial g}{\partial \eta}(u,v,T_I) \tag{32}$$

where $f_1(u,v)$ is the DEM, T_I refers to the transform $\boldsymbol{M}_{20}$ for image 2 or $\boldsymbol{M}_{10}$ for image 1, and the summation is over images 1 and 2. From (32), the derivative of g is

$$\begin{aligned}\frac{\partial g}{\partial \eta}(u,v,T_I) = \boldsymbol{R}'_{20}\frac{\partial f_2}{\partial \eta}(u',v') \\ + \frac{\partial \boldsymbol{R}'_{20}}{\partial \eta} f_2(u',v') + \frac{\partial \boldsymbol{t}'_{20}}{\partial \eta}.\end{aligned} \tag{33}$$

Using (31) and the fact that $\boldsymbol{R}_{21}$ and $\boldsymbol{t}_{21}$ are a constant matrix and vector provided by the iconic matching method, we can modify the about equation as

$$\begin{aligned}\frac{\partial g}{\partial \eta}(u,v,T_I) = \boldsymbol{R}'_{20}\frac{\partial f_2}{\partial \eta}(u',v') + \frac{\partial \boldsymbol{R}_{10}^{-1}}{\partial \eta}\boldsymbol{R}_{21}^{-1} f_2(u',v') \\ + \frac{\partial \boldsymbol{R}_{10}^{-1}}{\partial \eta}\boldsymbol{t}_{10} - \boldsymbol{R}_{10}^{-1}\frac{\partial \boldsymbol{t}_{10}}{\partial \eta} \\ + \frac{\partial \boldsymbol{R}_{10}^{-1}}{\partial \eta}\boldsymbol{R}'_{21}\boldsymbol{t}_{21}.\end{aligned} \tag{34}$$

We can compute analytically all the derivatives appearing in (34) except the dertivative of $f_2(u',v')$. However, we can actually

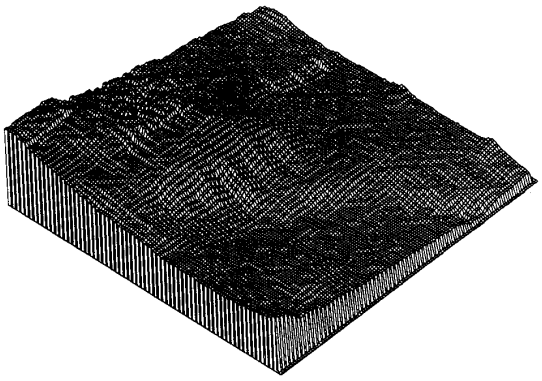


Fig. 24. Digital elevation map.

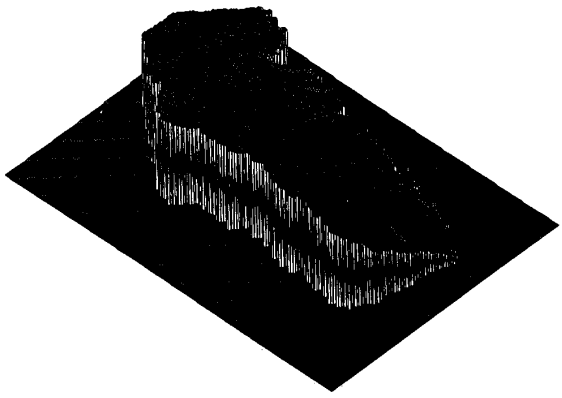


Fig. 25. Composite terrain map from multiple range images.

compute the derivative of $f_2(u', v')$ with respect to η by the chain rule from the range image 2 as described in Section III-B.

At each iteration, the algorithm

1) computes $g(u', v')$ with the generalized locus method
   - for range image 1, by applying the updated transformation $\boldsymbol{M}_{10}$ to the range measurements
   - for range image 2 by applying the updated transformation $\boldsymbol{M}_{20}$ to the range measurements
2) computes the error between the elevation of DEM and the two measurements from range images 1 and 2 by (20)
3) computes the partial derivative of the error function with respect to each of the transformation parameters by (32) in which we substitute T_I by
   - $\boldsymbol{M}_{10}$ for range image 1
   - $\boldsymbol{M}_{20}$ for range image 2
4) updates the transformation parameters for $\boldsymbol{M}_{10}$ by (21) and computes the transform $\boldsymbol{M}_{20}$ by compounding the updated transform $\boldsymbol{M}_{10}$ with the constant transform $\boldsymbol{M}_{21}$.

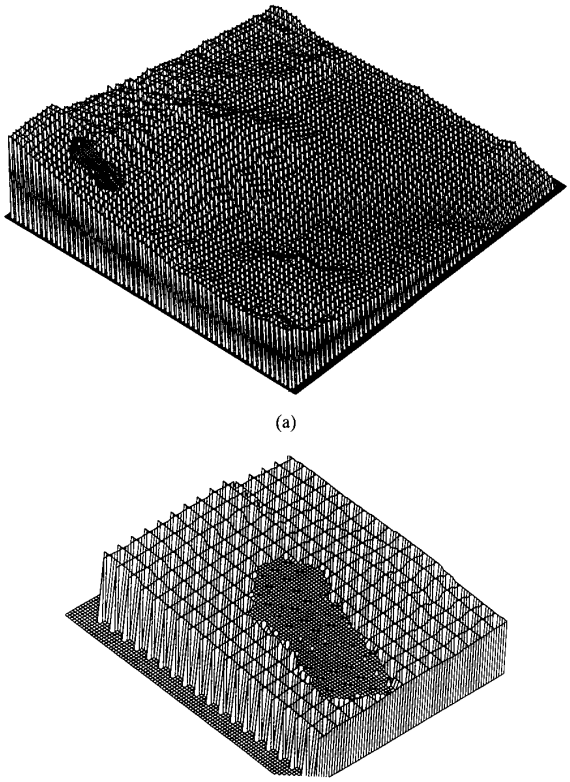
As in the local iconic matching method, iteration stops when the variation of error ΔE is small enough (convergence) or E itself is small enough.

4) Experimental Results: We have tested our algorithm for matching range images with a DEM using data collected by Hughes AI group on the ALV test site (multiple range images and a DEM) [19]. The DEM has 64 by 64 samples covering approximately 0.1 km^2.

Fig. 24 shows the DEM used in the experiments. A sequence of ERIM range images is independently obtained by using a mobile robot. Fig. 25 shows a composite terrain map built from those range images by using the local terrain matching methods described in Section III.

Fig. 26 shows the result of DEM matching with the range images. Fig. 26(a) shows the entire DEM overlaid with the composite terrain map from 30 range images. The resulting (or refined) DEM consists of two areas with different resolution: a) areas scanned by 30 range images, as shown in the upper left corner of Fig. 26, cover approximately 20 by 60 m with the resolution of 20 cm and b) areas covered by the original DEM has the resolution of 5 m. Fig. 26(b) shows the portion of DEM along the vehicle path. At the boundary between the DEM and the composite terrain map, we do not see any large discrepancies between elevations of the two maps.

However, as we increase the number of range images beyond 30, we have observed substantial discrepancies between the elevation values from range images and the DEM. We found that this was due to an invalid assumption that the DEM is perfectly reliable.



(a)

(b)

Fig. 26. Results for DEM matching.

When a human operator built a DEM from stereo pairs of aerial photographs, he introduced error in elevations for slopes due to his misperception when compiling slopes in alternating directions [4]. In the current DEM, we observe slopes after 30 images, and the error in elevations for those slopes is approximately 5 m. Since the average feature size in real terrain is smaller than 5 m error values, we simply cannot have any meaningful comparisons between the DEM and range images. Moreover, the estimates of vehicle roll were unavailable for this set of range images. We expect that this problem can be solved by improving the DEM as well as the initial estimate of robot position in the DEM.

VI. Conclusions

This work demonstrates the feasibility of a 3-D vision system for modeling rugged terrain. With such a system, mobile robots operating in rugged environments will be able to build accurate terrain models and confidently plan and execute appropriate actions. We have developed the locus method for perceiving and modeling rugged terrain from multiple sensor data.

Since the locus method operates directly in image space, it is much more straightforward than methods that operate in Cartesian space. Significant advantages include 1) creating elevation maps of arbitrary resolution, 2) converting sensor uncertainty into elevation map uncertainty, 3) easily and explicitly identifying shadow (occluded) areas, and 4) computing the upper bound of elevation in shadow areas.

Elevation values are computed using dense and uniform range measurements in image space instead of sparse range measurements in Cartesian space. Furthermore, shadow areas are easily identified by finding areas in image space where depth (range) discontinuities exist between neighboring pixels. In a similar manner, the locus method can compute upper bounds on terrain elevation within invisible (shadow) areas. It is not possible to compute these upper bounds using Cartesian-based approaches.

Beyond inherent simplifications from operation in image space, we have shown that the locus method can be applied to 1) building an accurate elevation map from a single range image, 2) estimating the incremental displacement between successive images, 3) building a composite terrain map for a large area from a long sequence of images, and 4) merging images from different sensors.

The locus method is a single general and powerful method that accepts data from various sensors and acquisition resolutions and efficiently builds terrain maps at any desired mapping resolution. Experimental results on real and synthetic data demonstrate that the locus method accurately merges data such as long sequences of range images. Furthermore, results indicate that the locus method can be used to merge a composite terrain map to a DEM. Using these various terrain models, mobile robots should be able to plan more confidently than with existing terrain models.

The locus method has been extensively applied to the analysis of range images acquired by a 3-D range finder. Future work can extend the locus method to various sensors such as 1-D laser scanners, dense stereo depth maps, light stripe range finders, color images, and thermal images. The locus method will be able to build an intermediate representation (e.g., an elevation map) from any depth maps and fuse any nondepth data into the representation. We expect that this extension would eventually provide a framework for low-level sensor fusion as we have already shown in fusing range and color data [12].

Acknowledgment

The authors would like to thank W. Whittaker, M. Hebert, and E. Krotkov for their helpful discussions throughout this work. We would also like to thank K. Olin of Hughes Research Laboratories for providing range images and the DEM. 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 United States Government.

References

[1] J. Bares, *et al.*, "An autonomous rover for exploring mars," *Special Issue Comput. Mag. Autonomous Intelligent Machines,* vol. 44, pp. 75–145, Oct. 1988.

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

[3] M. Brady, J. Ponce, A. Yuille, and H. Asada, "Describing surfaces," in *Proc.* CVGIP, Mar. 1985, pp. 1–28.

[4] D. L. Edwards, G. B. Desmond, and M. W. Schoppmann, "Terrain data base generation for autonomous land vehicle navigation," *Photogrammetria,* vol. 43, pp. 101–107, Apr. 1988.

[5] D. B. Gennery, "Visual terrain matching for a Mars rover," in *Proc. 1989 IEEE Conf. Robotics Automat.,* May 1989.

[6] W. E. L. Grimson, "Computational experiments with a feature-based stereo algorithm," *IEEE Trans. Patt. Anal. Machine Intell.,* vol. 7, no. 1, pp. 17–34, Jan. 1985.

[7] M. Hebert, C. Cailas, E. Krotkov, I. Kweon, and T. Kanade, "Terrain mapping for a roving planetary explorer," in *Proc. 1989 IEEE Conf. Robotics and Automat.,* May 1989.

[8] M. Hebert, T. Kanade, and I. S. Kweon, "3-D vision techniques for autonomous vehicles," Tech. Rep. CMU-RI-TR-88-12, Carnegie-Mellon Univ., Pittsburgh, PA, 1988.

[9] M. G. Kendall and P. A. P. Moran, *Geometrtical Probabilities.* New York: Hafner, 1963.

[10] I. Kweon, "Modeling rugged terrain by mobile robots with multiple sensors," Ph.D. thesis, Carnegie Mellon Univ., Robotics Institute, Pittsburgh, PA, Aug. 1990.

[11] I. Kweon, M. Hebert, and T. Kanade, "Perception for rugged terrain," in *Proc. SPIE Mobile Robots,* (Cambridge, MA), 1988.

[12] ——, "Sensor fusion of range and reflectance data for outdoor scene analysis," in *Proc. Space Oper. Automat. Robotics,* (Cleveland), 1988.

[13] I. Kweon, R. Hoffman, and E. Krotkov, "Experimental characterization of the perceptron laser rangefinder," Techn. Rep. CMU-RI-TR-91-1, Robotics Inst., Carnegie Mellon Univ., Pittsburgh, PA, 1991.

[14] P. M. Lee, *Bayesian Statistics: An Introduction.* Oxford: Oxford University Press, 1989.

[15] L. Matthies, R. Szeliski, and T. Kanade, "Kalman filter-based algorithms for estimating depth from image sequences," in *Proc. Image Understanding Workshop,* (Cambridge), 1988.

[16] J. E. W. Mayhew and J. P. Frisby, "Psychophysical and computational studies towards a theory of human stereopsis," *Artificial Intell.,* vol. 17, pp. 349–408, Aug. 1981.

[17] F. H. Moffitt and H. Bouchard. *Surveying.* New York: Harper Row, 1982.

[18] F. R. Norvelle, "Interactive digital correlation techniques for automatic compilation of elevation data," in *Proc. ASP/ACSM Conf.,* Feb. 1981.

[19] K. E. Olin, M. Daily, J. Harris, and F. M. Vilnrotter, "Knowledge-based vision technology overview for obstacle detection and avoidance," in *Proc. IU Workshop,* 1989.

[20] K. S. Roberts, "A new representation for a line," in *Proc. Comput. Vision Patt. Recogn.,* Ann Arbor, MI, 1988.

[21] R. Szeliski, "Bayesian modeling of uncertainty in low-level vision," Ph.D. thesis, Carnegie Mellon Univ., 1988.

[22] ——, "Estimating motion from sparse range data without correspondence," in *Proc. Int. Conf. Comput. Vision,* (Tarpon Springs, FL), Dec. 1988.

[23] D. Terzopoulos, "Multiresolution computation of visible-surface representations" Ph.D. thesis, Mass. Inst. Technol., Cambridge, 1984.

[24] H. J. Thomas, D. S. Wettergreen, C. E. Thorpe, and R. M. Hoffman, "Simulation of the Ambler environment," in *Proc. the 23rd Pittsburgh Conf. Modeling Simulation,* (Pittsburgh), 1990.