Proceedings of the 1992 IEEE/RSJ International Conference on Intelligent Robots and Systems Raleigh, NC July 7-10, 1992

Terrain Mapping for Long-Duration Autonomous Walking

Regis Hoffman and Eric Krotkov

School of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213-3890 U.S.A.

Abstract— **We describe the perception system for the Ambler, an autonomous, legged mobile robot that operates in rugged environments. The Ambler's perception system uses 3-D laser range images to construct elevation maps of the local terrain, processing hundreds of images and thousands of elevation points during a walking experiment. This paper presents the design, implementation, and analysis of a perception system to meet the needs of long-duration walking, emphasizing issues of performance, accuracy, and reliability.**

I. Introduction

The Ambler is an autonomous, walking robot that operates in rugged terrain [2]. The mechanism consists of two stacks, each with a set of three legs. The Ambler body carries on-board computing and control electronics; on top of the body is mounted a laser scanner that acquires 3-D data of the local terrain (Figure 1).

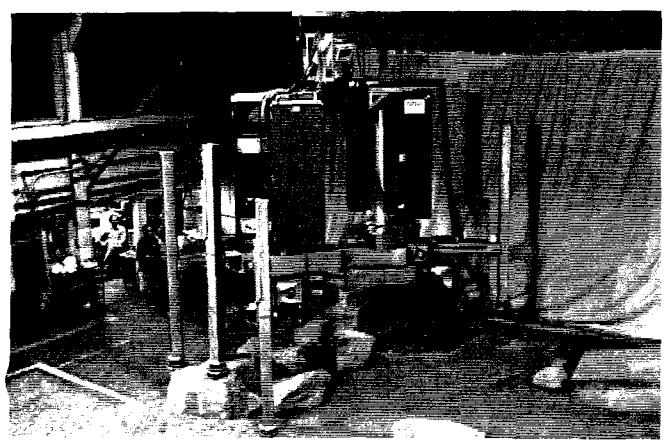

Fig. 1. The Ambler Robot.

The three major components of the Ambler's software system are perception, planning, and control. The perception system builds elevation maps of the local terrain from 3-D range images. The planning system uses elevation maps to determine obstacle-free routes and to select footfall locations, and the control system takes commands from the planner and executes body and leg trajectories.

Many researchers have advanced robotic perception along the frontiers of real-time performance (*e.g.* FastNav with a laser scanner [10], Dickmanns with multiple sensors [4], and Borenstein [3] with sonar) and high accuracy (*e.g.* Fleck [5] and Asada [1]). We believe that system reliability is as important as performance and accuracy. Long-duration (> 5 hour) autonomous walking requires a perception system that can meet three critical goals:

- *Performance* - The perception system should compute maps as quickly as possible, and must not be the bottleneck in the complete system. With a walking cycle time of about 1 minute, the elevation maps needed for one step must be computed in less than 30 seconds.

- *Accuracy* - The accuracy of the elevation maps (their consistency with ground truth), should remain nearly constant over time. Thus the perception system must minimize errors due to sensor thermal drift or dead-reckoning errors.

- *Reliability* - The system must handle thousands of map requests and process hundreds of images without fail. Special care must be taken to stay within the memory limits of the perception computers.

The first version of the perception system had shortcomings due to inaccuracies in the computed elevation maps and errors in the system design [6]. Inaccuracies in the map were both spatial (large elevation spikes and incorrect elevation values within a small region), and temporal (divergence of meter-sized terrain patches from the ground truth over time). Missing features in the system design became apparent during the nearly 100 hours of walking experiments. These included lack of modularity, difficulty in adjusting parameters during a walking experiment, and degradation of system performance over time.

We discuss the design, implementation, and performance analysis of the second-generation perception system. We present statistics of practical concern including memory utilization, processing times, and I/O bandwidth.

II. Perception System

The perception system builds terrain elevation maps of the local 3-D environment. The elevation maps are computed from range images acquired by a Perceptron laser scanner [7] that provides 256 x 256 pixel range and reflectance images, with 12 bits of data per pixel, at a frame rate of 2 Hz. The scanner has a 60 degree field of view in the horizontal and vertical directions, and an operating range of 2-40 meters. Figure 2 shows a pair of range and reflectance images from the Ambler test site.

0-7803-0737-2/92$03.00 1992©IEEE

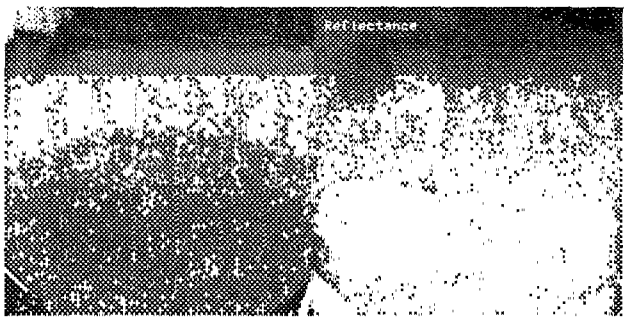

Fig. 2. Range (left) and Reflectance (right) Image.

The scene consists of a sand base with 1-2 meter high boulders (foreground) and a wooden ramp (background).

The perception system takes a sequence of range images I_0, I_1, ..., I_N together with the position of the robot body $B_0, B_1, ..., B_N$ (in a world reference frame). Given this set of range images and the position of the robot when the images were taken, we compute a map of the environment by merging maps created from each single image. Maps can be computed in any reference frame. Common frames include the world frame for global maps, the current body frame for leg recovery, and future body frames for advance planning. If a point in the map cannot be computed (due to invalid range data, outside the field of view, or occlusion), the map point is tagged as *unknown*.

Maps are computed from a sequence of range images, because maps created from a single frame of data do not, in general, contain enough information to accomplish even simple tasks. For example, consider the task of planning the trajectory of a recovering leg. Because the scanner looks forward, the map constructed from a single forward-looking image cannot possibly see obstacles either below or behind the vehicle.

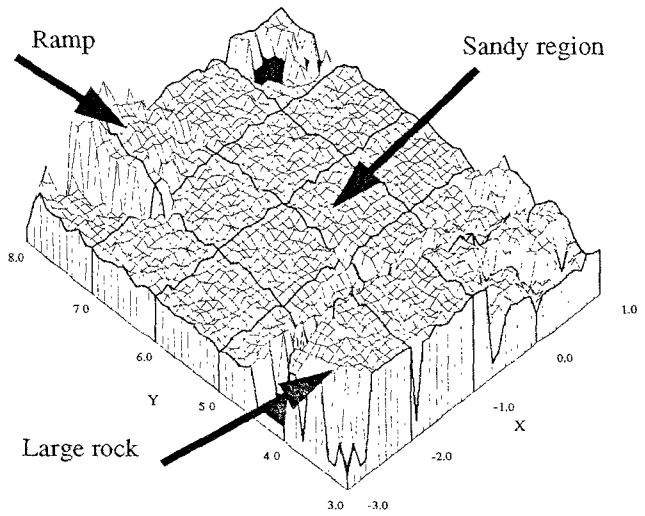

Fig. 3. Terrain Elevation Map.

Figure 3 shows a 4 x 5 meter elevation map at 10 cm resolution computed from the range image in Figure 2. The dark areas are labeled unknown regions, either because of terrain occlusion or bad data in the range image. The perception system only constructs elevation maps on demand from an external module (such as a planner). Typically the planner is interested in a small (< 1 meter square) terrain patch in front of the body. This small patch is called the *region of interest polygon* (ROI), and is expressed in some reference frame F. The perception system takes the region of interest polygon, plus a series of range images and produces a terrain elevation map within the region of interest polygon. This is represented pictorially in Figure 4.

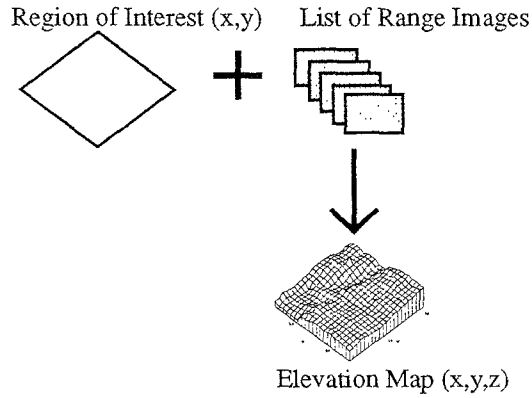

Fig. 4. Elevation Map for Region of Interest.

III. Design and Implementation

The perception system is a set of modules that communicate by sending and receiving messages through the Task Control Architecture (TCA), a central server that routes messages to and from communicating modules [9]. External modules query the perception system for elevation maps, and the perception system replies with the requested map.

Design goals of performance, map accuracy, and reliability, dictated several implementation decisions. As significant computation time is required to process the images and compute the maps, the perception system concurrently acquires images, processes them, and computes maps to improve performance. Also, because the range/reflectance image pair is rather large (256K), several mechanisms were added to reduce the amount of I/O bandwidth required between the perception modules.

Map accuracy is dominated by compounding errors in the robot's dead-reckoned position. Maps built in a world frame slowly diverge from ground truth over time because the robot's location is less well-known. To avoid the effects of compounding errors, the perception system can construct maps in the body frame of the robot.

Reliability implies that modules need not be restarted after they have begun execution. Memory constraints are severe; image storage alone can occupy all available memory. Careful deletion of images no longer required and re-using the memory

was an important implementation issue. From a practical standpoint, it is our experience that image thresholds, calibration parameters, and scale factors often vary slowly over time. It is necessary that these parameters be dynamically changeable during the perception system operation, either by the user, or preferably by another module, rather than by stopping and restarting a module.

A. Overall Design

The perception system consists of three major modules: the Perceptron Interface Module (PIM) acquires range images together with robot state information (body pose), the Image Queue Manager (IQM) processes and stores images, and the Local Terrain Module (LTM) constructs terrain elevation maps from the range image data on request from external modules.The modules communicate by sending and receiving messages - the PIM sends image data to the IQM, the LTM queries the IQM for range images, and the LTM is queried by the planner to build maps. Figure 5 summarizes the information flow between these modules.

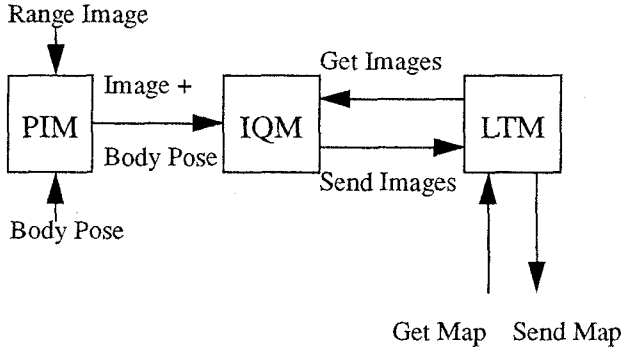

Fig. 5. Perception System Overview.

B. Perceptron Interface Module

Perceptron laser scanner range and reflectance images are acquired by the Perceptron Interface Module (PIM). Images can be either of type real (taken directly from the scanner), virtual (read from a file), or simulated (synthetic images generated by a simulation module). When a real image is taken, it is stamped with state information: the robot's configuration (body pose and leg positions), the scanner to body transformation matrix (obtained during a calibration phase), the sensor's field of view polygon, and the scale and offset parameters that transform a range sensor measurement into units of meters. As the PIM takes real Perceptron images, they can be optionally stored to disk files (together with the state information). These can later be used as virtual images by the PIM when the Perceptron scanner is unavailable.

Image acquisition by the PIM is triggered by the completion of a robot body move. At this time the body is stationary and the Ambler's legs are usually outside the laser scanner's field of view (reducing occlusions). The range and reflectance images together with the state information, are sent to the Image Queue Manager for processing and storage.

C. Image Queue Manager

Image and state information acquired by the PIM is sent to the Image Queue Manager (IQM). The IQM is responsible for processing the images (edge detection, connected region analysis, thresholding), queue manipulation (insert, delete, first, last), and responding to queries from other modules to return lists of images. Several image formats are supported, including range, reflectance, and black and white camera images. Because image data sets can be quite large, the IQM deletes old images after a maximum image limit is reached.

Each IQM image consists of two fields, *raw images* and *auxiliary images*. Raw images are unprocessed sensor data (*e.g.* range and reflectance images), while auxiliary images are optionally computed from raw images (*e.g.* edge image). Therefore, a point *(r, c)* in an auxiliary image gives some information about the corresponding point *(r, c)* in the raw image.

After a Perceptron image is received by the IQM, the following auxiliary images are computed:

- *Edge Image.* Produced by applying Canny operator over range image. Pixels are labeled EDGE or NO_EDGE.
- *Leg Image.* Predict where (if) the robot's legs are located in the range image. Label pixels as LEG or NO_LEG.
- *Ambiguity Image.* Apply connected region analysis to find range pixels outside ambiguity interval (due to sensor noise, or beyond the operating range of sensor). Label pixels as VALID or NOT_VALID.
- *Vignette Image.* Certain pixels near the lower corners of the range image are unusable due to a scanner aperture problem. These occur at a fixed location in the range image and are labeled NOT_VALID.
- *Good Pixel Image.* This image indicates if the corresponding range pixel should be used in map computation, and is a composite of the leg, ambiguity, and vignette image. A GOOD pixel has leg image = NO_LEG and ambiguity image = VALID and vignette image = VALID, otherwise it is labeled BAD.

Figure 6 shows the results of this processing, with a range image (upper left), reflectance image (upper right), the edge image (lower left), and good pixel image (lower right - good pixels white, bad pixels black). Bad pixels are caused by the two legs in the image, the scanner aperture (lower corners), and some noisy range data in the upper portion of the image. Typically 80-90% of the range pixels are labeled as good.

After an image is stored in the queue, it can be accessed by other modules. Because in general, a sequence of range images is needed to build a map, the IQM also provides lists of images to external modules.

Fig. 6. Raw and Auxiliary Images.

D. Local Terrain Module

Local terrain maps are constructed by the Local Terrain Module (LTM) from range images. Map computation is normally initiated by the planning system, which queries the LTM for an elevation map within a polygonal region (the region of interest polygon). Maps can be built in either a world frame, the current body frame, or a future body frame (to allow advance planning of leg moves). The LTM first obtains range images stored by the IQM, then uses the *locus method* [8] to transform the input range images into an output elevation map. A point within the map can have several possible states:

- *Known:* Elevation is computed and stored.
- *Unknown:* Impossible to compute elevation (point outside field of view, no valid image pixels).
- *Shadow:* Point is occluded, but upper bound on elevation is stored.

Because maps used by the planner are computed in body frame, they are not merged into a composite map. Rather, a new map is built each time a request is received.

E. Concurrent Operation

Each module in the perception system executes concurrently on a SUN workstation. Figure 7 shows the concurrent operation of image acquisition, image processing and map computation.Timing experiments show that the PIM takes about 1 second to take an image and body pose (on a SUN 3/E). The IQM image processing takes about 10 seconds (on a SUN SPARC 2), and the LTM needs 8 seconds to compute a 1 meter square elevation map (on a SUN SPARC 2).

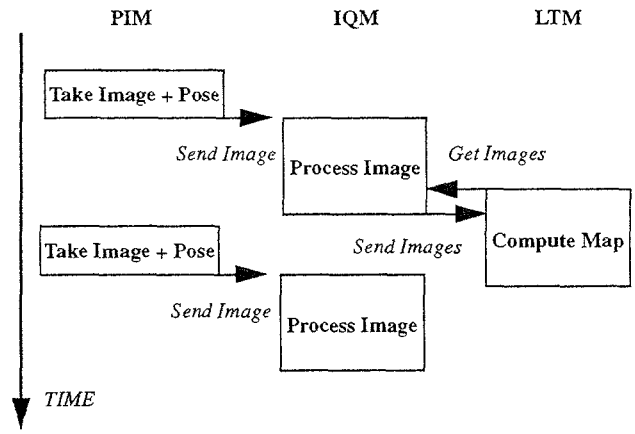

Fig. 7. Concurrent Operation.

IV. Results

During autonomous operation of the Ambler, statistics were kept of the perception system performance, accuracy, and reliability. In the longest continuous walking experiment, the robot travelled 107 meters in a period of 7.4 hours. During this run, the perception system took 400 range images and built 1300 elevation maps (600,000 elevation points). The following sections chart and analyze the statistics from this run.

A. Performance

The issue of performance is paramount because the large size of images (256K for Perceptron range and reflectance images and 128K for the auxiliary images) makes it impractical to transfer the entire image list each time the LTM needs range images from the IQM. We note the following:

1. In a list of N range images, only a subset P is required to compute an elevation map.
2. Within a range image, only a fraction of the image field of view is needed (typical map size ~$1m^2$, image field of view ~ $20m^2$).

The graph in Figure 8 illustrates assertion 1. The y axis shows the number of range images needed to build a 50 cm^2 map as a function of time. The number of images required varies between 4 (straight-line trajectory, flat terrain) and 28 (point turn, obstacles). Figure 9 shows a frequency distribution of the data from Figure 8. In a majority of the cases, only 5 range images are required to build a map; however in certain instances up to 28 images are used in building the terrain elevation map.

Figure 10 is the frequency distribution of the percentage of the total range image pixels used in the map computations. For each elevation map, the number of range image pixels used in computing the map was recorded. This justifies assertion 2, because we note that only 3-25% of a single range image is used in building the small, meter-sized maps requested by the planning system.

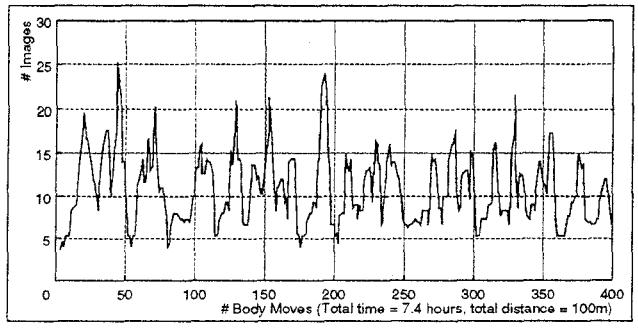

Fig. 8. Number of Range Images Needed to Build a Map.

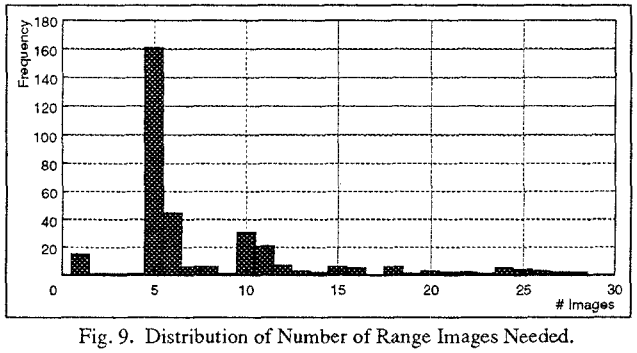

Fig. 9. Distribution of Number of Range Images Needed.

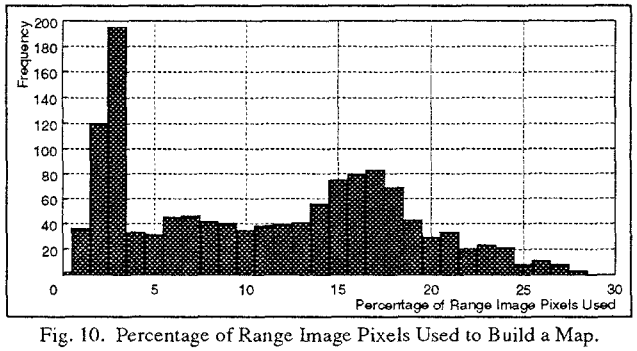

Fig. 10. Percentage of Range Image Pixels Used to Build a Map.

From these observations of the perception system behavior, enhancements reduced the amount of image data transferred between the IQM and LTM. Given a region of interest polygon, the IQM:

- Computes the list of images whose field of view intersects the region of interest.
- Compute the portion or *subimage* of each image that intersects the region of interest.
- Returns this list of subimages to the LTM for map computation.

The IQM thus provides only subimages that intersect the region of interest polygon. This significantly reduces the amount of data transfer between the IQM and LTM.

B. Accuracy

Accuracy is the error or difference between the ground truth and the computed elevation map. As the Ambler walks over the terrain, ground truth underneath the advancing foot is measured by the Ambler leg vertical displacement: the elevation map error is the ground truth minus the elevation computed by the perception system. Figure 11 is a plot of the elevation error over the test run.

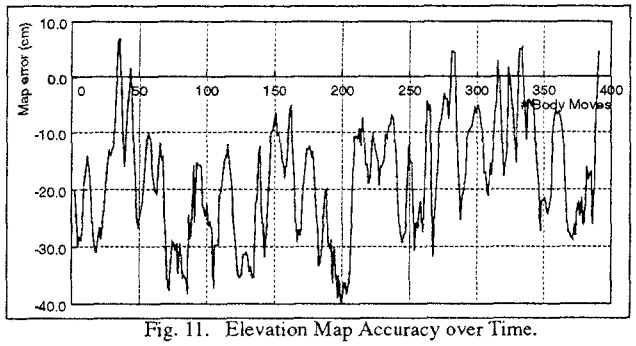

Fig. 11. Elevation Map Accuracy over Time.

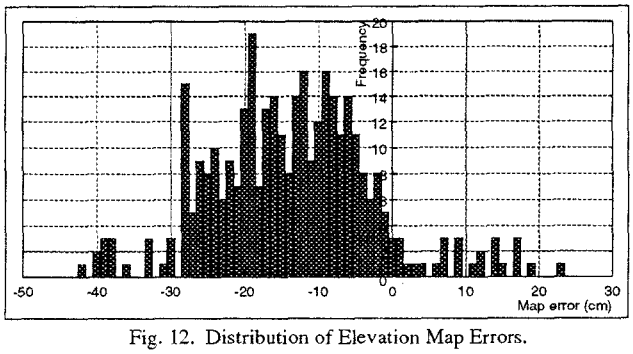

Fig. 12. Distribution of Elevation Map Errors.

Figure 12 shows the frequency distribution of elevation map errors. The mean error is -18.2 cm with a standard deviation of 10.3 cm (this compares to the scanner precision of 10-15 cm). We could make the mean error zero by the addition of a constant term, but we prefer this negative bias. The negative bias implies that perception maps are higher than ground truth, and this adds a safety margin to leg moves over the terrain.

C. Reliability

The perception system reliability is dominated by memory use. Images and maps consume significant amounts of memory. Long-term operation requires that limits be placed on memory use and that storage be reclaimed and re-used by deleting un-needed images and maps. Figure 13 shows the perception system's memory use as a function of the number of body moves. After 400 body moves, the memory requirements peak at 2.5 M (PIM), 2.8 M (LTM), and 30 M (IQM).

An additional aspect of reliability is the ability to change system parameters during the course of an experiment. Rather than restarting a module with a new parameter value, each module can be sent a message with parameter adjustment information. For example, if an edge detection threshold value changes (perhaps due to a change in the ambient light), the user (or another module) can send the IQM a message to adjust the edge threshold without the need to restart the IQM. This increases system reliability.

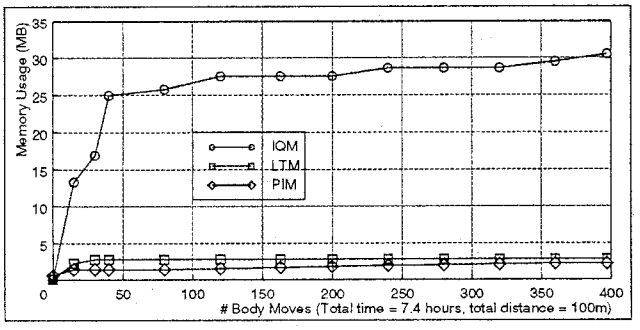

Fig. 13. Perception System Memory Use.

V. Conclusions

In our longest continuous walking experiment to date, the robot travelled 107 meters in a period of 7.4 hours. Average walking speed, including all computation, is 40cm/min (each body move is about 50cm) with motion of the mechanism the main limitation to the speed. During operation, the real-time control system is active about 80% of the time, while the planners and perception subsystems are each active about 50% of the time, and the centralized Task Control module is active only about 3% of the time (the total is greater than 100% because operations occur concurrently).

During this run, the perception system took 400 range images, computed and transferred 20,000 subimages, and built 1300 elevation maps (600,000 elevation points). At each step the planner queries for three maps. Each step requires about 1700 map computations, 4-28 range images to build the map, and 20-30 seconds time for map computation and I/O.

The goals of performance and reliability were met. Computation time for the maps is within the limit of 30 seconds (50% of the walking cycle). System memory use remains relatively constant after about 100 steps. Accuracy is acceptable, but can be improved. Current work is investigating the use the error in the elevation maps (difference in ground truth measured by legs and the computed elevation map) as an error signal in a perception system feedback loop.

Acknowledgments

The authors thank the members of the Ambler team for assistance in conducting walking experiments. This includes Brian Albrecht, Christopher Fedor, Henning Pangels, Reid Simmons, and David Wettergreen. This work was supported by the National Aeronautics and Space Administration under Grant NAGW-1175.

References

[1] M. Asada. Map Building for a Mobile Robot from Sensory Data. *IEEE Transactions on Systems, Man, and Cybernetics.* 20(6), November-December, 1990.

[2] J. Bares, M. Hebert, T. Kanade, E. Krotkov, T. Mitchell, R. Simmons and W. Whittaker. Ambler: An Autonomous Rover for Planetary Explorations. *IEEE Computer.* 18-26, June, 1989.

[3] J. Borenstein and Y. Koren. Histogramic In-Motion Mapping for Mobile Robot Obstacle Avoidance. *IEEE Transactions on Robotics and Automation.* 7(4), August, 1991.

[4] E.D. Dickmanns, B. Mysliwetz, and T. Christians. An Integrated Spatio-temporal Approach to Automatic Visual Guidance of Autonomous Vehicles. *IEEE Transactions on Systems, Man, and Cybernetics.* 20(6), November-December, 1990.

[5] M. Fleck. A Topological Stereo Matcher. *International Journal of Computer Vision.* August, 1991.

[6] R. Hoffman and E. Krotkov. Perception of Rugged Terrain for a Walking Robot: True Confessions and New Directions. *Proceedings of the International Workshop on Intelligent Robots and Systems.* IEEE, November, 1991.

[7] I.S. Kweon, R. Hoffman and E. Krotkov. *Experimental Characterization of the Perceptron Laser Rangefinder.* Robotics Institute Technical Report CMU-RI-TR-91-1, Carnegie Mellon University, 1991.

[8] I.S. Kweon. *Modeling Rugged Terrain by Mobile Robots with Multiple Sensors.* Ph.D. thesis, Robotics Institute, Carnegie Mellon University, January, 1991.

[9] R. Simmons. An Architecture for Coordinating Planning, Sensing, and Action. *Proceedings of DARPA Planning Workshop.* San Diego, CA, November, 1990.

[10] S. Singh and P. Keller. Obstacle Detection for High Speed Autonomous Navigation. *Proceedings of the IEEE International Conference on Robotics and Automation.* IEEE, Sacramento, California, April, 1991.