1990 Year End Report
Autonomous Planetary Rover at Carnegie Mellon

William Whittaker, Takeo Kanade, and Tom Mitchell
Principal Investigators

CMU-RI-TR-91-19

The Robotics Institute
Carnegie Mellon University
Pittsburgh, Pennsylvania 15213

August 1991

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

Contents

List of Figures

Abstract

This report describes progress during 1990 at the Robotics Institute, Carnegie Mellon University, in research on a six-legged autonomous robot, called the Ambler, designed for planetary exploration. The report summarizes the achievements during calendar year 1990, and lists personnel and publications. Also included are several papers that describe significant aspects of the research.

Much of the research in 1990 focused on understanding the capabilities of the robot itself and developing a software system capable of autonomously walking the Ambler through natural terrain. We also investigated algorithms for perceiving and acquiring rock samples, which were demonstrated on a separate testbed.

Understanding Mechanism Capabilities — Characteristics important to mobile robots in general, and walking robots in particular, include their mobility, stability, and dynamic performance. We investigated each of these aspects using a combination of mathematical and geometrical analyses, simulation, and experiments using robotic mechanisms. Results largely confirmed our expectations about the Ambler's high degree of mobility, rigidity, and large tipover safety margin.

Autonomous Walking — This area combined four distinct efforts: real-time control of the Ambler, perception of rugged terrain using laser range scanning, algorithms for planning leg and body moves, and task-level control to integrate and coordinate the various components. We embarked on a comprehensive set of experiments, both in simulation and on the Ambler, that systematically added new components and increased the complexity of the environment traversed. These efforts culminated at year end in a successful demonstration to NASA consisting of some 25 autonomous steps walking on sandy, rolling terrain and crossing boulders, a wooden ramp, and stepping out of a meter deep trench. In total, the Ambler autonomously walked over a kilometer in 1990.

Sample Acquisition — In preparation for demonstrating a complete mission scenario, we developed methods for perceiving the three dimensional shape of rocks in natural, cluttered surroundings using a light-striper. The shape descriptions were used to plan and execute grasp procedures to collect the rocks. A graphical user interface enabled humans to interact with the system, aid in the selection of rocks, and monitor progress. The methods were demonstrated using the Robot World testbed, and proved quite robust in practice.

Introduction

This report reviews progress from January through December 1990 at the Robotics Institute, Carnegie Mellon University, on research sponsored by NASA titled "Autonomous Planetary Rover." This section presents an overview of the project and a summary of accomplishments during the year. It also lists members of the research group and publications by the group relevant to the Planetary Rover project. The report also includes reprints of four papers that highlight significant aspects of the research.

Overview

The CMU Planetary Rover project has built and is currently testing the Ambler, a six-legged robot designed for planetary exploration. We are interested in studying general capabilities of walking machines, including their mobility, stability, and dynamic performance. The Ambler has a novel configuration consisting of a stacked arrangement of orthogonal legs (Figure 1). It features a unique *circulating gait* in which trailing legs recover past all others to become leading legs. The circulating gait significantly decreases the average number of steps needed for travel. The Ambler has been designed to stably traverse a 30° slope while crossing meter sized surface features (e.g., ditches, boulders, and steps).

We have developed perception, planning, real-time control, and task-level control algorithms for the Ambler. The perception subsystem uses data from a scanning laser rangefinder to autonomously calibrate the rangefinder and to build 3D maps of the terrain. The planning subsystem combines kinematic constraints on the robot's motion with terrain constraints to find leg and body moves that provide good forward progress and stability. The real-time control coordinates the Ambler's joints to perform accurate leg and body moves, maintains the dead-reckoned position, and monitors the status of the robot. The task-level control facilitates concurrent operation of the subsystems, reliable execution monitoring and error recovery, and management of the Ambler's computational and physical resources.

The component subsystems have been integrated into a system that autonomously walks the Ambler through obstacle-strewn terrain. Our chief objectives here are to enable the Ambler to traverse extremely rugged terrain and to operate for extended periods of time, both without human intervention. This involves developing reliable and efficient components, and incorporating many levels of safeguards, both in hardware and software.

While much of our experimental program involves the six-legged Ambler itself, we have utilized several other testbeds to develop parts of the system. These include the single-leg testbed [22, 31], 2D and 3D graphical simulators [60], a leveling testbed [49], and the Robot World testbed for investigating sample acquisition [12].

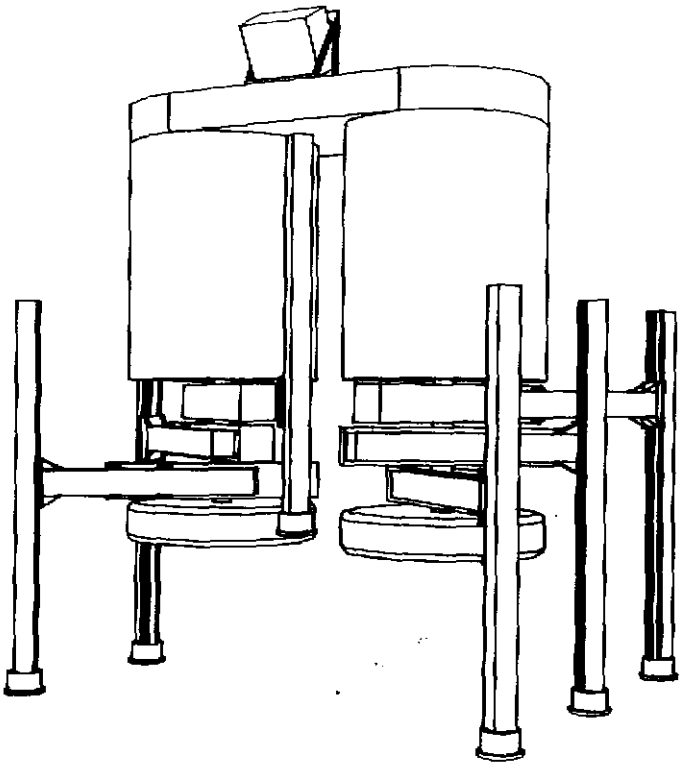


Figure 1: The Ambler Robot

Accomplishments

The major accomplishments of the Planetary Rover project during calendar year 1990 can be divided into three main areas: 1) understanding the capabilities of the Ambler mechanism, 2) autonomous walking, and 3) sample acquisition.

Understanding Mechanism Capabilities

The Ambler was designed to be a highly mobile walking machine. There were, however, certain aspects of its capabilities that required further analysis and development. John Bares, who designed the rover, continued his thesis research by developing objective criteria for comparing different walking mechanisms, concentrating on the advantages of using orthogonal legs and a stacked arrangement of the legs, which facilitates a circulating gait [5]. David Manko completed his PhD thesis entitled "A General Model of Legged Locomotion on Natural Terrain" [43] in which he developed methods for efficiently simulating the dynamical interactions between legged vehicles and the terrain. His analysis indicated the relative stability of the Ambler under servo control. The models were also used to demonstrate the rigidity of the mechanism (vertical deflections) and estimate load distributions. A planar dynamical model [45] was used to study the optimal choice of joints to be used for body propulsion.

Questions of the Ambler's stability received much attention. A study of the vehicle's tipover

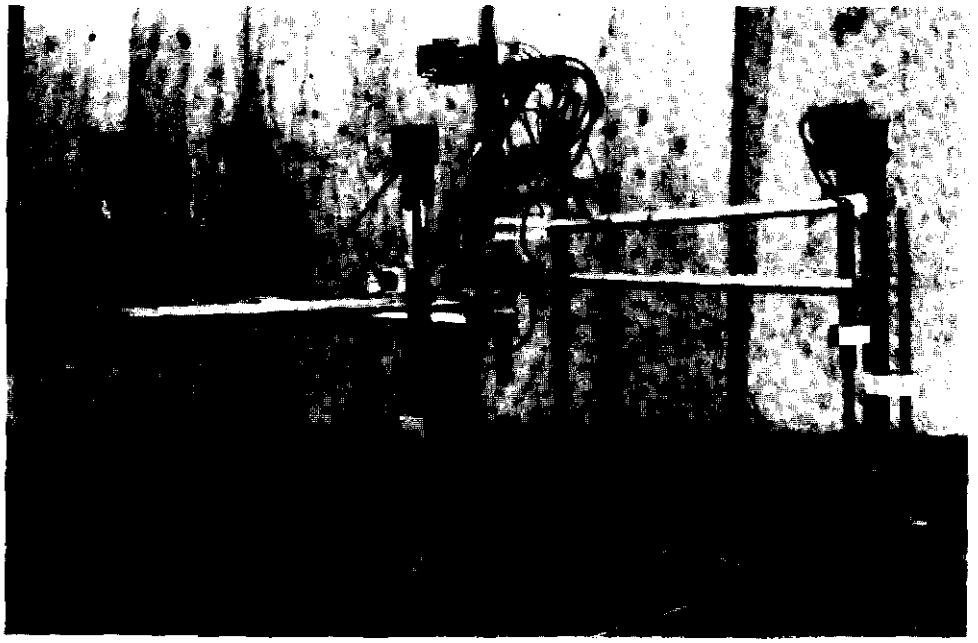


Figure 2: The Leveling Testbed

characteristics was undertaken. The study also produced an algorithm for a stability monitor that would determine the *energy stability margin* of the machine (the energy needed to cross the tipover point) and warn if margin was too close.

In the same vein, a comprehensive research effort was instituted to investigate methods for controlling the Ambler's attitude (tilt) and altitude (vertical height). Methods under investigation include active leveling, which performs real-time feedback control based on inclinometer readings, static leveling, based on calculating the vertical leg extensions needed to bring the machine to level, and force redistribution of the loads on the feet. The research employed the dynamical models developed by David Manko and a leveling testbed composed of six stacked vertical actuators (Figure 2). The leveling testbed demonstrated the use of static leveling to correct for accumulations of tilt, and active leveling to react to terrain failures. The static leveling algorithm was adapted for the Ambler and used during our autonomous walking experiments.

Studies were begun to measure the Ambler's power usage under various situations. Preliminary results indicate that the steady-state power consumption of the motors, amplifiers, and associated electronics for the 2500 kg Ambler is about 1800 watts. Lifting and moving a single leg takes an additional 150 watts, propelling the body horizontally takes 450 watts, and lifting the body uses an additional 1800 watts above steady state.

Autonomous Walking

We devoted much effort in 1990 to upgrading the Ambler mechanism and developing the component technologies of real-time control, perception, planning, and task control. The components

were integrated together into a system that successfully demonstrated autonomous walking over obstacle-strewn terrain.

Mechanism: Several upgrades to the mechanism were performed. Custom digital/analog multiplexors were completed and installed on all six legs to reduce the number of wires passing the Ambler's slip rings. Six axis force/torque sensors were installed on the feet and the signals were calibrated. Two inclinometers were mounted on the body structure to provide information about the tilt of the Ambler.

The most significant upgrade was a rework of the vertical link gearboxes. The as-built weight of the Ambler exceeded the design and necessitated a higher gearing ratio to provide the necessary torque. The new gear boxes are designed to provide sufficient torque even with the additional weight when the computing and power generation equipment are moved on-board.

Real-Time Control: The real-time controller was extended to provide generalized leg and body moves. Leg move commands are specified by a series of joint space waypoints, and the controller executes the trajectories so that all motions begin and end simultaneously. When moving legs, the controller enables interrupts in the force sensors to detect lift-off and ground contact. The controller will not move the leg horizontally until lift-off is detected, and immediately stops all motions when a preset force threshold is reached.

Body move commands are specified by a desired translation and rotation. The trajectory generated by the controller translates the body linearly, while simultaneously rotating about the vertical axis. The body move uses a trapezoidal velocity profile to smoothly accelerate and decelerate.

A kinematic dead-reckoning algorithm was implemented that determines the Ambler's position and orientation using the leg joint positions and inclinometer data. Given an initial stance, the positions of all feet in a global frame are computed. Following each body move the Ambler's location is estimated by finding the rotation matrix and translation vector that, with the least squared error, transform the positions of the feet in the vehicle frame to their stored positions in the global frame. This method of dead-reckoning is fairly accurate (typically within 1-2cm), and can often detect when individual feet have slipped.

Perception: A new scanning laser rangefinder, manufactured by Perceptron, was received. The scanner acquires data in 256x256 pixel images at a rate of 2Hz. It digitizes to 12 bits over approximately 40m, which provides a range resolution of 0.98cm. Extensive tests were conducted to characterize and calibrate the scanner. Major deficiencies were identified that required repairs by Perceptron.

An automatic procedure was implemented to find the transformation between the scanner frame and the Ambler body frame. The procedure moves the leg to various positions within the scanner field of view, and processes the reflectance image to locate the leg. The leg positions in

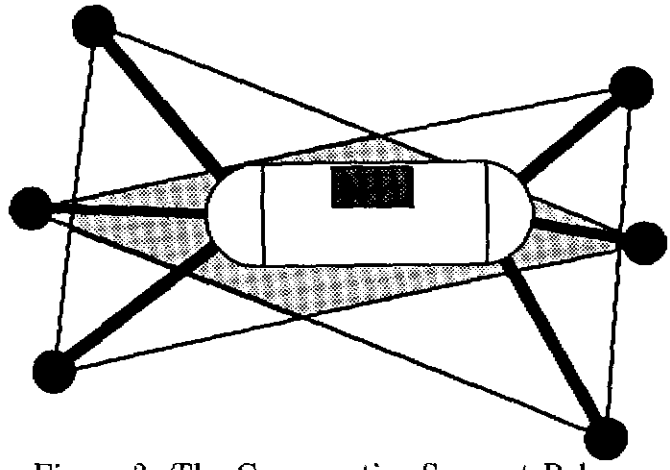


Figure 3: The Conservative Support Polygon

the images and those positions returned by the controller are used to compute the transformation that minimizes the measurement errors [24].

Extensive testing was performed on an algorithm for iconic matching of terrain maps, which determines the transformation between two viewing frames. While computationally expensive, this method could be used to augment dead-reckoning to provide a more accurate estimate of the Ambler's motion. A technique was implemented to compute iso-elevation contours from terrain maps to find qualitative features such as "ridges" and "ravines." This technique may be useful as input to higher-level route and trajectory planning algorithms.

The Single Leg Testbed was used to conduct experiments on terrain typing using force sensors. Different materials were stepped on and the forces plotted. An identification procedure was formulated to classify different materials based on stiffness and friction characteristics. The procedure successfully classified terrains consisting of sand, soil, and sawdust.

Planning: We have developed a hierarchy of planning algorithms for walking the Ambler through rough terrain: a gait planner, a footfall planner, and a leg recovery planner.

The gait planner finds leg and body moves to follow arcs of any radius [61]. The planner combines kinematic and pragmatic constraints on motion. Relevant kinematic constraints include limits on leg motion and the need to avoid collisions with other legs. Pragmatic constraints include not placing the leg in the path of the body, to avoid having to shuffle legs unnecessarily. For safety, the planner imposes the constraint that the Ambler's center of gravity must always remain within the *conservative support polygon (CSP)* (Figure 3), which ensures that the Ambler will remain stable even if any single leg fails.

The footfall planner is used to analyze the underlying terrain to determine locations that provide stable and tractive footfalls. While stability and traction cannot be measured directly from elevation maps, we can extract features from the maps that are good indicators, such as terrain slope, roughness, and curvature. The footfall planner combines these features using a neural net that is trained using human-supplied preferences on footfall locations [21].

The leg recovery planner produces energy efficient trajectories for the legs that avoid terrain features and collisions with the mechanism itself. The planner first determines a trajectory for

the horizontal joints and then optimizes the vertical trajectory to minimize travel time. This is accomplished by projecting an envelope of vertical locations created by assuming that the leg raises/lowers at full speed while traveling horizontally.

The three planners were tested extensively, both in simulation and on the Ambler itself. We have developed two graphical kinematic simulators: a 2D version that shows an overhead view of the Ambler and operates on Sun workstations, and a 3D closed-loop simulator operating on an Iris that, in addition to the Ambler model, includes simulated terrain and laser scanner [60]. These simulators have proven invaluable in designing and testing the planning code.

Task Control: Work on the Task Control Architecture concentrated in two areas: concurrency, and error detection and recovery. Using TCA features for task sequencing and monitoring, the single-leg walking system was modified to concurrently execute one step while planning the next. This improved the average walking speed by over 30%, with competence comparable to the original sequential system. These ideas were subsequently ported to the Ambler walking system, with similar improvements in walking speed.

Much attention was focused on using the monitoring and exception handling capabilities of TCA to detect execution errors and replan when necessary. These efforts had a significant impact on the reliability of the walking system. Work continues in this area in order to achieve long-term, continuous operation of the Ambler.

The TCA code was completely rewritten in 1990 in order to provide more modular and reliable utilities. In particular, the internal flow of control was documented and redesigned to provide a more maintainable and extensible system.

Walking Experiments: In 1990, we embarked on a comprehensive experimental program to test the Ambler and the walking system. The Ambler is housed in a large high-bay that is covered with sand and includes a wooden ramp, hillocks, trenches, and boulders (Figure 4).

Our experimental methodology involved incrementally adding modules in order to tackle increasingly difficult terrain. We began on a flat, non-compliant floor, manually operating the Ambler through the controller interface. Next, the gait planner was used to walk the Ambler along a variety of arcs, including straight-line paths and point turns.

To traverse obstacle-strewn terrain, the perception modules and leg recovery planners were added. The leg recovery planner used the maps produced from the range data images (Figure 5) to guide the Ambler's legs over and around terrain features.

Figure 6 shows a typical run of the Ambler. Operating in concurrent planning and execution mode, the system has an average walking speed of 35cm/min, of which 75% of the time is spent by the controller in actually moving the mechanism, with the planners and perception subsystems each active about 50% of the time. Figure 7 shows the module utilization for a typical run: the dark shaded areas of the chart are times when modules are computing; lightly shaded areas are when they are awaiting replies from other modules.

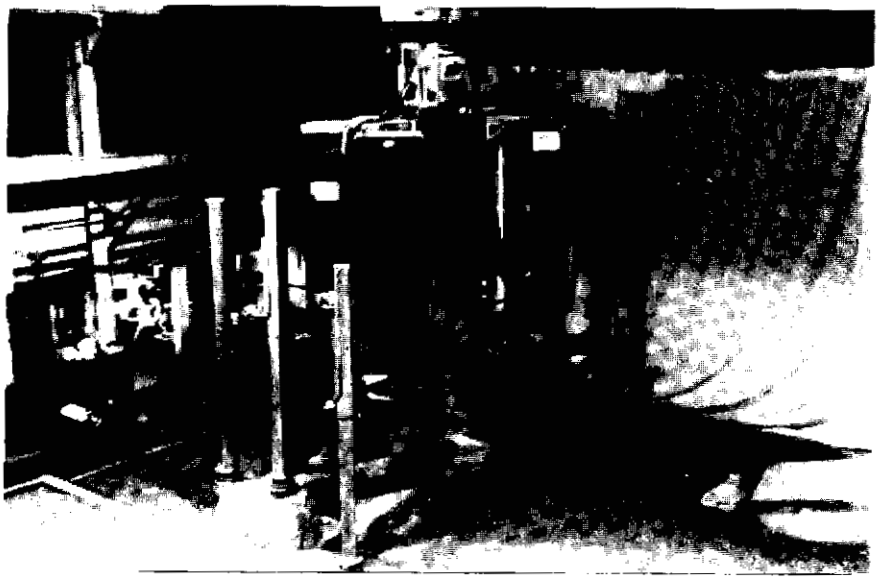


Figure 4: The Ambler Obstacle Course

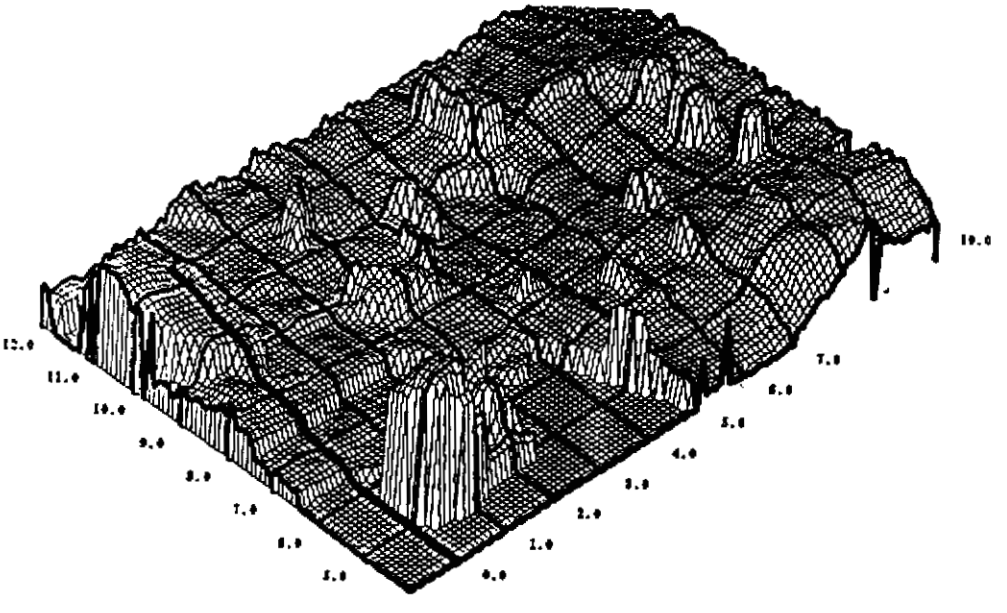


Figure 5: A Local Terrain Map

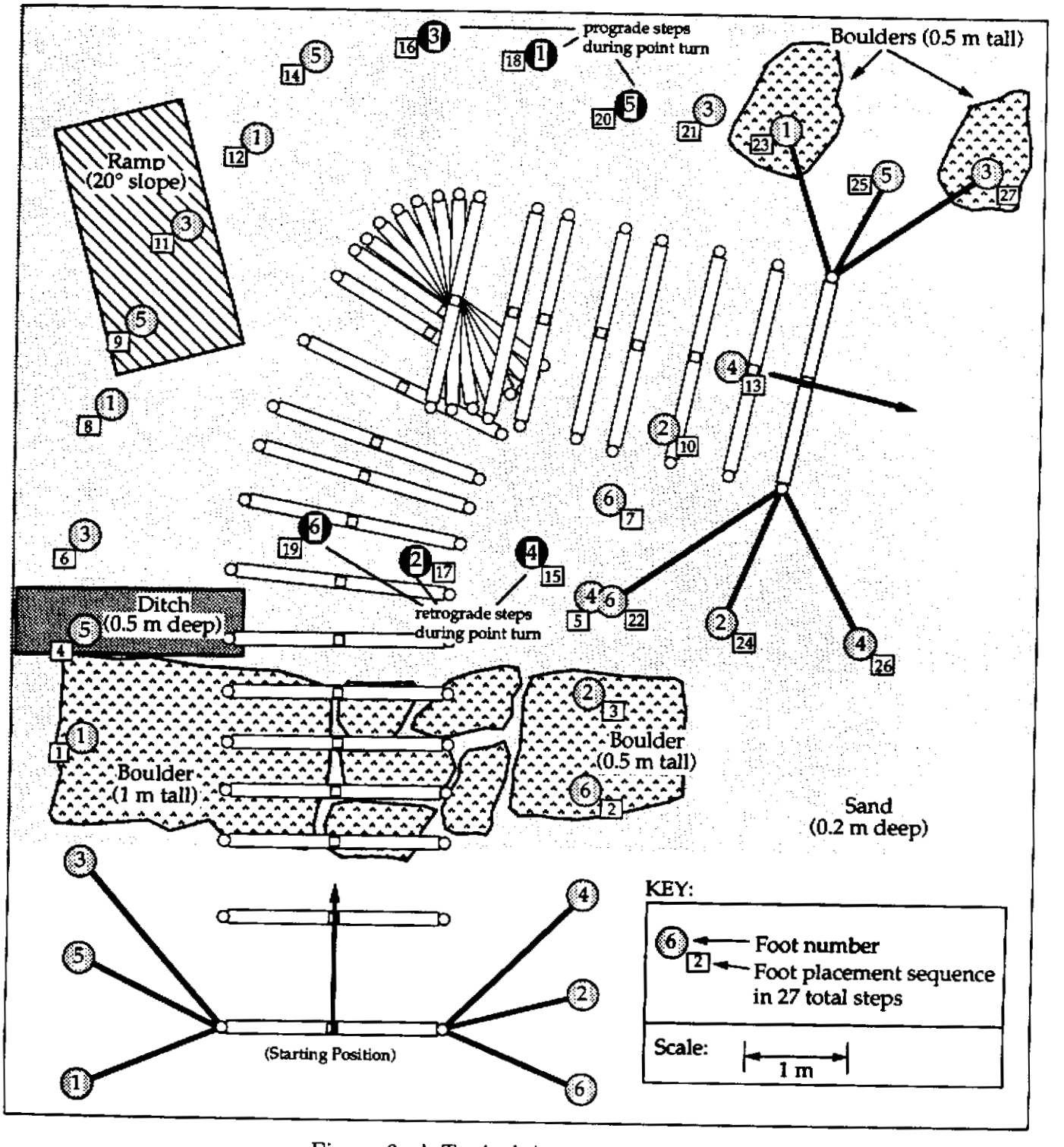


Figure 6: A Typical Autonomous Walk

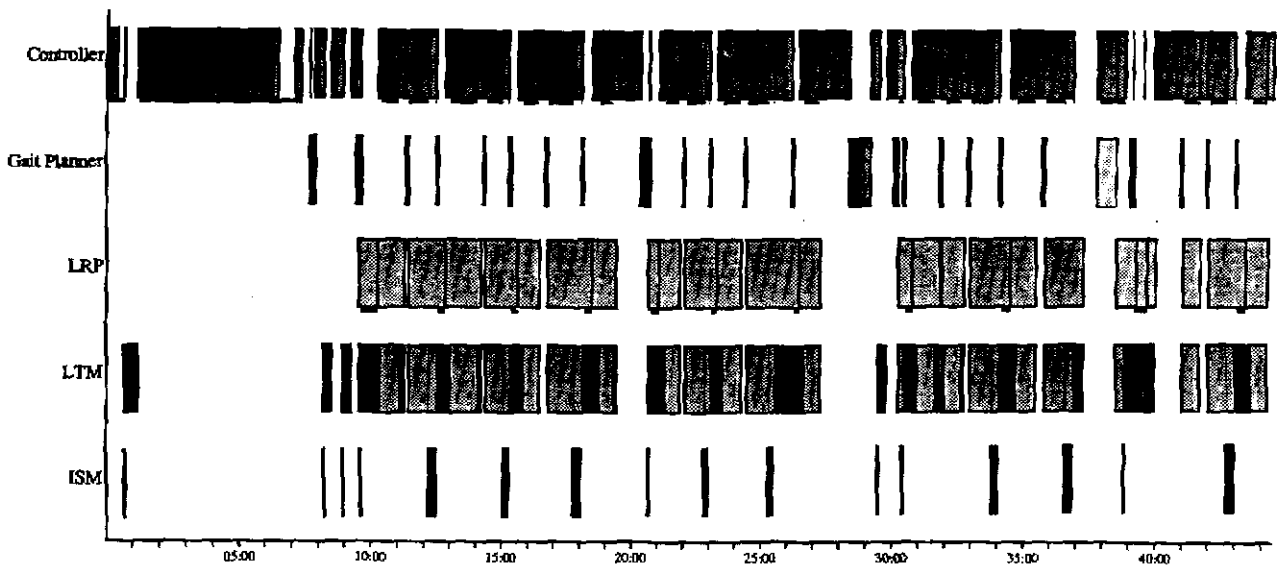


Figure 7: Timing Chart for a Typical Run

In total, the Ambler has walked autonomously well over a kilometer. While, for safety during testing, the Ambler is normally attached to a crane sling to catch it in case of mechanical or algorithmic failure, on occasion the Ambler has walked "slingless."

We have also used preplanned motions to test the limits of the Ambler's mobility. This has included stepping down the sheer face of a meter high rock into a meter deep trench, propelling with several feet poised on the edges of boulders, walking with one foot in the air (to demonstrate conservative support), and raising the body to its full height of six meters.

Sample Acquisition

A research effort was begun in 1990 to perceive and acquire rock samples in natural terrain, using the Robot World testbed. The testbed consists of an X/Y translation table, a light-striper to acquire range data, and a manipulation arm with custom gripper. Sand and rocks are placed within view of the light-striper and the arm (Figure 8).

Images from the light-striper are processed to extract features such as surface discontinuities. These are used to find object boundaries and contours. We developed several shape extraction algorithms, including one based on fitting superquadrics and one based on deformable contours. The deformable contours model proved to be more effective, since it makes fewer a priori assumptions about the shape of the rocks. The sampling perception system is able to extract sample targets from range images without human intervention.

The parameters of the object representation are then used to plan a grasping motion to acquire the rock. We developed planning algorithms for two types of grasps: an enveloping grasp

Figure 8: The Sample Acquisition Testbed

using a clam-shell gripper, and a pinch grasp using a three-fingered gripper. The enveloping grasp is used for picking up isolated rocks, and the pinch grasp is used in cluttered situations.

A graphical user interface was developed to assist in this project. The interface had an extensive selection of menus, which enabled researchers to customize algorithms interactively. The interface also showed the results of the various operations (feature extraction, object fitting, grasp selection, etc.).

The sample acquisition system was tested extensively in a variety of conditions — from scenes with isolated rocks, to jumbled piles of stones. The perception and planning parts of the system proved to be fairly robust over a wide range of conditions, although a more reliable gripper needs to be further investigated.

Personnel

The following personnel were directly supported by the project, or performed related and contributing research in 1990:

Faculty: Martial Hebert, Katsushi Ikeuchi, Takeo Kanade, Eric Krotkov, Tom Mitchell, Reid Simmons, Chuck Thorpe, William Whittaker.

Staff: Brian Albrecht, Purushothaman Balakumar, Gary Baun, Mike Blackwell, Kevin Dowling, Christopher Fedor, Kerien Fitzpatrick, Regis Hoffman, Jim Martin, Clark McDonald, Jim Moody, Dave Pahnos, Henning Pangels, Bill Ross, Kevin Ryan.

Visiting Scientists: Kenichi Arakawa, Claude Caillas, Herve Delingette, Fabrice Noreils, Bao Xin Wu.

Graduate Students: John Bares, Lonnie Chrisman, Richard Goodwin, Goang Tay Hsu, In So Kweon, Long-Ji Lin, David Manko, Peter Nagy, Gerry Roston, Ming Tan, Jay West, David Wettergreen.

Undergraduate Students: Steve Baier, Jonathan Burroughs, Doug DeCarlo, John Greer, Nathan Harding, Andy John, Terry Lim, Eric Miles, Hans Thomas.

References

[1] J. Bares. Orthogonal Legged Walkers for Autonomous Navigation of Rugged Terrain. Ph.D. Thesis Proposal, Department of Civil Engineering, Carnegie Mellon University, December 1988.

[2] J. Bares, M. Hebert, T. Kanade, E. Krotkov, T. Mitchell, R. Simmons, and W. Whittaker. Ambler: An Autonomous Rover for Planetary Exploration. *IEEE Computer*, 22(6):18–26, June 1989.

[3] J. Bares and W. Whittaker. Configuration of an Autonomous Robot for Mars Exploration. In *Proc. World Robotics Conf. on Robotics Research: The Next Five Years and Beyond*, pages 37–52, Gaithersburg, Maryland, May 1989.

[4] J. Bares and W. Whittaker. Ambler: A Walking Robot for Autonomous Planetary Exploration. In *Proc. International Symposium on Artificial Intelligence, Robotics, and Automation in Space*, Kobe, Japan, November 1990.

[5] J. Bares and W. Whittaker. Walking Robot with a Circulating Gait. In *Proc. IEEE Intl. Workshop on Intelligent Robots and Systems*, pages 809–818, Tsuchiura, Japan, July 1990.

[6] J. Bares and W. Whittaker. Orthogonal Legged Walking Robot. United States Patent. Allowed 1990., To be awarded 1991.

[7] C. Caillas. Imaging Sensing to Identify Footfall Positions for a Legged Robot. In *Second Workshop on Military Robotic Applications*, Kingston, Ontario, August 1989. Royal Military College of Canada and Civil Institute of Environmental Medicine.

[8] C. Caillas. Autonomous Robot Using Infrared Thermal Camera to Discriminate Objects in Outdoor Scene. In *Proc. SPIE Conf. Applications of Artificial Intelligence VIII*, Orlando, Florida, April 1990.

[9] C. Caillas. Thermal Imaging for Autonomous Vehicle in Outdoor Scenes. In *Proc. IEEE Intl. Workshop on Intelligent Robots and Systems*, pages 651–658, Tsuchiura, Japan, July 1990.

[10] C. Caillas. Thermal Imaging for Robotic Applications in Outdoor Scenes. Technical Report CMU-RI-TR-90-8, Robotics Institute, Carnegie Mellon University, April 1990.

[11] C. Caillas, M. Hebert, E. Krotkov, I. S. Kweon, and T. Kanade. Methods for Identifying Footfall Positions for a Legged Robot. In *Proc. IEEE Intl. Workshop on Intelligent Robots and Systems*, pages 244–250, Tsukuba, Japan, September 1989.

[12] T. Choi, H. Delingette, M. DeLuise, Y. Hsin, M. Hebert, and K. Ikeuchi. A Perception and Manipulation System for Collecting Rock Samples. In *Proc. NASA Symposium on Space Operations, Applications, and Research*, Albuquerque, New Mexico, June 1990.

[13] C. Fedor, R. Hoffman, G. Roston, and D. Wettergreen. Software Standards and Guidelines. Technical Report PRWP-89-2, Robotics Institute, Carnegie Mellon University, 1989.

[14] C. Fedor and R. Simmons. Task Control Architecture User's Manual. Technical Report PRWP-89-1, Robotics Institute, Carnegie Mellon University, 1989.

[15] P. R. Group. Experiments in Perception, Planning, and Control for the Carnegie Mellon Mars Rover. Robotics Institute, Carnegie Mellon University, unpublished working document, December 1988.

[16] P. R. Group. Integration of Perception, Planning, and Control in the Carnegie Mellon Mars Rover. Robotics Institute, Carnegie Mellon University, unpublished working document, December 1988.

[17] M. Hebert, T. Kanade, E. Krotkov, and I. S. Kweon. Terrain Mapping for a Roving Planetary Explorer. In *Proc. IEEE Intl. Conf. on Robotics and Automation*, pages 997–1002, Scottsdale, Arizona, May 1989.

[18] M. Hebert, T. Kanade, and I. Kweon. 3-D vision techniques for autonomous vehicles. Technical Report CMU-RI-88-12, The Robotics Institute, Carnegie Mellon University, 1988.

[19] M. Hebert, E. Krotkov, and T. Kanade. A Perception System for a Planetary Explorer. In *Proc. IEEE Conf. on Decision and Control*, pages 1151–1156, Tampa, Florida, December 1989.

[20] R. Hoffman and E. Krotkov. Terrain Roughness Measurement from Elevation Maps. In *Proc. SPIE Conf. on Advances in Intelligent Robotics Systems, Mobile Robots IV*, pages 104–114, Philadelphia, Pennsylvania, November 1989.

[21] G. T. Hsu and R. Simmons. Learning footfall evaluation for a walking robot. In *Proceedings of Machine Learning Workshop*, Chicago, IL, June 1991.

[22] T. Kanade, T. Mitchell, and W. Whittaker. 1988 Year End Report: Autonomous Planetary Rover at Carnegie Mellon. Technical Report CMU-RI-TR-89-3, Robotics Institute, Carnegie Mellon University, January 1989.

[23] E. Krotkov. Active Perception for Legged Locomotion: Every Step is an Experiment. In *Proc. IEEE Intl. Symposium on Intelligent Control*, pages 227–232, Philadelphia, Pennsylvania, September 1990.

[24] E. Krotkov. Laser Rangefinder Calibration for a Walking Robot. Technical Report CMU-RI-TR-90-30, Robotics Institute, Carnegie Mellon University, Pittsburgh, Pennsylvania, December 1990.

[25] E. Krotkov. Qualitative Perception of Shape and Material. In *Proc. AAAI Workshop on Qualitative Vision*, pages 209–213, Boston, Massachusetts, July 1990.

[26] E. Krotkov. Laser Rangefinder Calibration for a Walking Robot. In *Proc. IEEE Intl. Conf. Robotics and Automation*, pages 2568–2573, Sacramento, California, April 1991.

[27] E. Krotkov, J. Bares, M. Hebert, T. Kanade, T. Mitchell, R. Simmons, and W. Whittaker. Design of a Planetary Rover. *1988 Annual Research Review, The Robotics Institute, Carnegie Mellon University*, pages 9–24, 1989.

[28] E. Krotkov, J. Bares, M. Hebert, T. Kanade, T. Mitchell, R. Simmons, and W. Whittaker. Ambler: A Legged Planetary Rover. *1990 Annual Research Review, The Robotics Institute, Carnegie Mellon University*, pages 11–23, 1991.

[29] E. Krotkov, C. Caillas, M. Hebert, I. S. Kweon, and T. Kanade. First Results in Terrain Mapping for a Roving Planetary Explorer. In *Proc. NASA Conf. on Space Telerobotics*, pages 247–256, Pasadena, California, January 1989. Jet Propulsion Laboratory Publication 89-7, Vol. 2.

[30] E. Krotkov, G. Roston, and R. Simmons. Integrated System for Single Leg Walking. Technical Report PRWP-89-3, Robotics Institute, Carnegie Mellon University, 1989.

[31] E. Krotkov, R. Simmons, and C. Thorpe. Single Leg Walking with Integrated Perception, Planning, and Control. In *Proc. IEEE Intl. Workshop on Intelligent Robots and Systems*, pages 97–102, Tsuchiura, Japan, July 1990.

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

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

[34] I. S. Kweon. Modeling Rugged 3-D Terrain from Multiple Range Images for Outdoor Mobile Robots. Ph. D. Thesis Proposal, School of Computer Science, Carnegie Mellon University, July 1989.

[35] I. S. Kweon, M. Hebert, and T. Kanade. Perception for Rugged Terrain. In *Proc. SPIE Mobile Robots III Conf.*, Cambridge, Massachusetts, November 1988. Society of Photo-Optical Instrumentation Engineers.

[36] I. S. Kweon, M. Hebert, and T. Kanade. Sensor Fusion of Range and Reflectance Data for Outdoor Scene Analysis. In *Proc. NASA Symposium on Space Operations, Applications, and Research, NASA Conf. Publication 3019*, pages 373–382, Dayton, Ohio, July 1988.

[37] I. S. Kweon and T. Kanade. High Resolution Terrain Map from Multiple Sensor Data. In *Proc. IEEE Intl. Workshop on Intelligent Robots and Systems*, pages 127–134, Tsuchiura, Japan, July 1990.

[38] L.-J. Lin, T. Mitchell, A. Phillips, and R. Simmons. A Case Study in Robot Exploration. Technical Report CMU-RI-TR-89-1, Robotics Institute, Carnegie Mellon University, January 1989.

[39] L.-J. Lin, R. Simmons, and C. Fedor. Experience with a Task Control Architecture for Mobile Robots. Technical Report CMU-RI-TR-89-29, Robotics Institute, Carnegie Mellon University, 1989.

[40] S. Mahalingam. Terrain Adaptive Gaits for the Ambler. Master's thesis, Department of Mechanical Engineering, University of North Carolina, 1988.

[41] S. Mahalingam and W. Whittaker. Terrain Adaptive Gaits for Walkers with Completely Overlapping Leg Workspaces. In *Proc. Robots 13*, pages 1–14, Gaithersburg, Maryland, May 1989.

[42] D. Manko. Models of Legged Locomotion on Natural Terrain. Ph.D. Thesis Proposal, Department of Civil Engineering, Carnegie Mellon University, June 1988.

[43] D. Manko. *A General Model of Legged Locomotion on Natural Terrain.* PhD thesis, Dept. of Civil Engineering, Carnegie Mellon University, April 1990.

[44] D. Manko and W. Whittaker. Inverse Dynamic Models used for Force Control of Compliant Closed-Chain Mechanisms. In *Proc. ASME Design and Automation Conf.*, pages 61–66, Montreal, August 1989.

[45] D. Manko and W. Whittaker. Planar Abstraction of a Prototype Walking Machine. In *Proc. 20th Pittsburgh Conf. on Modeling and Simulation*, pages 1817–1823, Pittsburgh, Pennsylvania, May 1989.

[46] P. Nagy. Attitude and Altitude Control for a Novel 6-Legged Robot. Research Prospectus, Mechanical Engineering Department, Carnegie Mellon University, January 1989.

[47] P. Nagy. Motion Control for Multi-Legged Walking Vehicles on Rugged Terrain. Ph.D. Thesis Proposal, Dept. of Mechanical Engineering, Carnegie Mellon University, June 1990.

[48] P. Nagy and W. Whittaker. Experimental Program for the CMU Mars Rover Single Leg Testbed. In *Proc. 20th Pittsburgh Conf. on Modeling and Simulation*, pages 1825–1829, Pittsburgh, Pennsylvania, May 1989.

[49] P. Nagy and W. Whittaker. Motion Control for a Novel Legged Robot. In *Proc. IEEE Symp. Intelligent Control*, pages 2–7, Albany, New York, September 1989.

[50] R. Simmons. An Architecture for Building and Controlling Mobile Robots. In *Proc. Workshop on Domain-Specific Software Architectures*, Hidden Valley, Pennsylvania, July 1990.

[51] R. Simmons. An Architecture for Coordinating Planning, Sensing, and Action. In *Proc. DARPA Workshop on Planning*, San Diego, California, November 1990.

[52] R. Simmons. Concurrent Planning and Execution for a Walking Robot. Technical Report CMU-RI-TR-90-16, Robotics Institute, Carnegie Mellon University, July 1990.

[53] R. Simmons. Robust Behavior with Limited Resources. In *Proc. AAAI Spring Symposium*, Stanford, California, March 1990.

[54] R. Simmons and E. Krotkov. Perception, Planning and Control for Walking on Rugged Terrain. In *Proc. NASA Symposium on Space Operations, Applications, and Research*, Albuquerque, New Mexico, June 1990.

[55] R. Simmons and E. Krotkov. An Integrated Walking System for the Ambler Planetary Rover. In *Proc. IEEE Intl. Conf. Robotics and Automation*, pages 2086–2091, Sacramento, California, April 1991.

[56] R. Simmons, E. Krotkov, and G. Roston. Integrated System for Single Leg Walking. Technical Report CMU-RI-TR-90-15, Robotics Institute, Carnegie Mellon University, Pittsburgh, Pennsylvania, July 1990.

[57] R. Simmons, L.-J. Lin, and C. Fedor. Autonomous Task Control for Mobile Robots. In *Proc. IEEE Intl. Symposium on Intelligent Control*, pages 663–668, Philadelphia, Pennsylvania, September 1990.

[58] R. Simmons and T. Mitchell. A Task Control Architecture for Autonomous Robots. In *Proc. NASA Symposium on Space Operations, Applications, and Research*, Houston, Texas, July 1989.

[59] R. Simmons and T. Mitchell. A Task Control Architecture for Mobile Robots. In *Proc. AAAI Spring Symposium*, Stanford, California, March 1989.

[60] H. Thomas, D. Wettergreen, C. Thorpe, and R. Hoffman. Simulation of the Ambler Environment. In *Proc. 23rd Pittsburgh Conf. Modeling and Simulation*, May 1990.

[61] D. Wettergreen, H. Thomas, and C. Thorpe. Planning Strategies for the Ambler Walking Robot. In *Proc. IEEE Intl. Conf. Systems Engineering*, August 1990.

[62] W. Whittaker, T. Kanade, and T. Mitchell. 1989 Year End Report: Autonomous Planetary Rover at Carnegie Mellon. Technical Report CMU-RI-TR-90-4, Robotics Institute, Carnegie Mellon University, February 1990.

IEEE International Workshop on Intelligent Robots and Systems
IROS '90

WALKING ROBOT WITH A CIRCULATING GAIT

John E. Bares and William L. Whittaker

Field Robotics Center, Carnegie Mellon University
Pittsburgh, Pennsylvania 15213, USA

Abstract: Mobile robots have yet to navigate and explore rough terrains autonomously. The Ambler walking robot has been developed in response to this challenge, specifically autonomous exploration of planetary and lunar surfaces. The Ambler, which maintains its body on a level trajectory while walking, has unique orthogonal legs that are stacked under the body instead of the traditional animal-like arrangement of legs around the body. The resulting stacked configuration leads to a novel circulating gait that promises to improve mobility in terrains of extreme roughness. The Ambler's level body motion, orthogonal legs, and circulating gait greatly simplify physical control, terrain model construction, and motion planning–all of which are traditional impediments to autonomous travel across rough terrain.

1. INTRODUCTION

Rough terrains that occur in mining, construction, and planetary surfaces are vast irregular landscapes of rock, soil, and sand. The surface of Mars (Fig. 1) is a prototypical rough, natural terrain. Such terrain would challenge many animals and defy most mechanical locomotion. Exploration of rough terrain without continuous human monitoring and with only occasional human interaction (termed autonomous exploration) is an unrealized ideal for mobile robots. An autonomous robot fails if it enters a situation from which it cannot reverse–the need for extreme reliability and safety affects the entire system, from mechanical configuration to planning strategy. Unlike their teleoperated counterparts, autonomous explorers are controlled by sensor-driven planning algorithms. If the robot configuration does not permit a feasible strategy for planning and perception, or if the robot is not physically suited to rough terrain operation, competent autonomy is unrealizable.

Fig. 1: Surface of Mars

This paper describes the Ambler[1] a prototype terrestrial walker for rough terrain, and discusses the major advantages of the Ambler's circulating gait configuration in the context of autonomous exploration criteria.

An autonomous rough terrain exploration robot must reliably transport payloads across the terrain. Rough terrain can be characterized by:

- Three-dimensional geometry including boulder-covered steep slopes, steps, and ditches. In addition to being a locomotion challenge, this geometry creates occlusions for terrain sensors. As a result, some areas in the terrain cannot be observed and should be avoided.
- Materials such as sand and dust whose strength and stability for vehicle support cannot be confidently determined by non-contact sensing.

The combination of these factors can cause much of a rough terrain's surface area to be unacceptable for positive, confident vehicle support. Acceptable

1 AMBLER is an acronym for Autonomous, MoBILe, Exploration Robot.

areas for vehicle support may be sparse and non-contiguous.

Straight line traverse of rough terrain is insufficient for exploration: As new information learned during exploration modifies paths and intentions, a robot must respond to ever changing plans. It should be capable of general motion in rough terrain, including turning, reversing, and moving laterally. The planning required to provide the robot with general motion capability should be feasible to implement with available computational equipment. Since exploration robots are far ranging and generate their own power, high payload to weight ratio and extreme efficiency pose additional requirements. An exploring robot's configuration should also strongly consider deployment needs of the payload. For instance, a sampling drill may need to be positioned accurately against the terrain and supported rigidly during operation.

2. WALKING LOCOMOTION

2.1 Advantages of Walking for Rough Terrain

Mobile robot locomotion candidates include mechanisms that roll, walk, or combine rolling and walking (Track laying mechanisms are grouped here with rolling mechanisms because tracks are in continuous terrain contact and are analogous to large wheels.). The fundamental difference between a rolling and walking mechanism is the means by which terrain support is provided. Wheeled machines have rollers in continuous contact with the terrain, while walkers suspend themselves over the terrain on discrete contact points. Because a walker only requires few support areas (footholds) compared to the continuous path required for a wheel, a walker can traverse otherwise impassable terrain. A walking machine isolates its body from the underlying terrain and smoothly propels its body independent of terrain details and foot placement. Large body lifts with respect to gravity (e.g, climbing a step) are difficult moves for a wheeled machine because the motion is accomplished by shearing the terrain–commonly resulting in large energy losses and unstable motion. A walker can smoothly and stably lift and lower its body while maintaining vertical foot forces. Walking machines conserve power in rough terrain in part because the body can be maintained at a fairly constant orientation and elevation with respect to gravity. Additionally, power losses to terrain are through discrete instead of continuous deformations.

A vehicle's support polygon is defined by the lines connecting the points of ground support when projected on a ground plane (plane orthogonal to the gravity vector). Stability can be described as the distance from the vehicle's center of gravity projected on the ground plane to the polygon edge in the direction of interest. Most rolling vehicles have little control over their stability. Alternately, a walker can move its body (i.e., center of gravity) with respect to its ground support points (feet) to affect its stability.

Walkers are quite amenable to the deployment needs of scientific and sampling payloads. Smooth, stable body motion is ideal for sensor perspectives on the terrain. With its feet in fixed locations, a walker can precisely move its body three-dimensionally to position body-mounted instruments and tools over, on, and into the terrain surface.

Walking was selected as an advantageous means of locomotion for rough terrain exploration. Walkers are able to succeed with few acceptable terrain support areas, body-terrain isolate, and provide unique benefits to scientific and sampling payloads. Furthermore, legged machines can optimize their stability and conserve power during traversal of rough terrains.

2.2 Terrain Adaptive Walkers

Perhaps the simplest walkers that can traverse rough terrain are frame-type walkers. An example is the Komatsu underwater octopod ReCUS, which consists of two rectangular frames, each with four vertically telescoping legs–the machine walks over rough terrain by supporting itself on one four-legged frame while advancing the other frame (4). The ReCUS steers by rotating the lifted frame with respect to the supporting frame. Since groups of legs are advanced and placed together, a frame walker's capability is limited in terrain where many areas are deemed unsuitable for footholds.

Unlike frame walkers, terrain adaptive walkers place feet individually–each foot can be moved in three dimensions to conform to the terrain. The Adaptive Suspension Vehicle (ASV) (11), Aquarobot (5), and Odex I (9) are terrain adaptive walkers. Terrain adaptive walkers are able to select high confidence footholds and optimize stability even in very rough terrains. The difficulty of complete freedom of foot motion is that selection of individual footholds in rough terrain requires intensive modeling and planning. The ASV lessens

this burden by relying on its human operator to select footholds and plan body moves through rough terrain.

An alternate method that reduces the complexity of foothold selection is to use precomputed templates that automatically place feet in patterns with little or no consideration for the specifics of the underlying terrain. An alternating tripod gait, used by the Odex I, Aquarobot, and other six-leg walkers is an example. In this gait, legs support and advance in sets of three. However, since patterned groups of acceptable footholds are required, the capability of this gait in rough terrain is limited. Furthermore, during each advance cycle the vehicle's safety is quite vulnerable as failure of any one of the three supporting legs would destabilize the walker.

3. AMBLER

The Ambler (Fig. 2 and Table 1) is a prototype rough terrain robot that is responsive to the basic needs of autonomous exploration of planetary and lunar surfaces. Terrains considered for Ambler mobility included slopes up to 30° with frequent surface features (e.g., ditches, boulders, and steps) of up to one meter in size. Design payloads consisted of scientific and sampling equipment such as tooling for grasping, digging, and deep coring (several meter). The power budget was taken as one kilowatt for travel speed of 1 meter/minute.

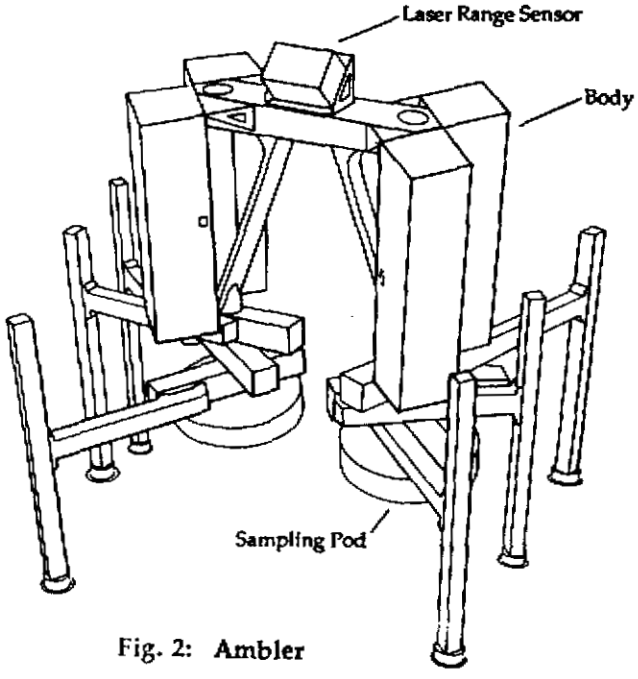


Fig. 2: Ambler

Each Ambler leg consists of a rotational and extensional link that move in the horizontal plane and an orthogonal vertical link. The Ambler's six "orthogonal" legs are stacked on two central body shafts (three legs to a shaft). The two shafts are connected to an arched body structure that includes four enclosures that house power generation, electronics, computing and scientific equipment.

A laser range sensor mounted atop the body builds terrain maps from which footholds are selected. Two sampling "pods" under the leg stacks can accommodate large sampling tools or sensors and can be placed in near proximity to or directly on the terrain. Deep coring equipment could be housed in the central body shafts that extend the full height of the walker. The Ambler's structural elements are primarily aluminum construction.

3.1 Ambler Operation

The Ambler's vertical links individually adjust to terrain roughness and maintain the walker in a continuous level orientation over the terrain (Fig. 3). Equal displacements on all vertical links lift or lower the body to climb or descend slopes, steps, etc. Propulsion of the level body is achieved by coordinated motions of the rotational and extensional links. Passive foot rotation allows the vertical links to pivot about the feet during propulsion.

Table 1: Ambler Specifications

Dimensions	
Typical walking width:	4.5 m
Typical walking length:	3.5 m
Typical foot spacing in direction of motion:	1.5 m
Height:	4.1-6.0 m
Mobility	
Maximum step crossing:	1.9 m
Maximum ditch crossing:	1.5 m
Maximum slope with 1 m wide ditch on slope:	30°
Weight [2]	
Legs and body structure:	2050 kg
Vehicle design weight (with payload):	3180 kg
Power [3]	
Body propulsion at 2 m/min:	650 watts
Leg recovery:	150 watts

[2] Because Ambler functionality is essential for experimentation and work in many other program areas, some compromises were made with respect to weight and efficiency to guarantee basic walking performance.

[3] Not included: computing and steady state power (power consumption when walker is not moving).

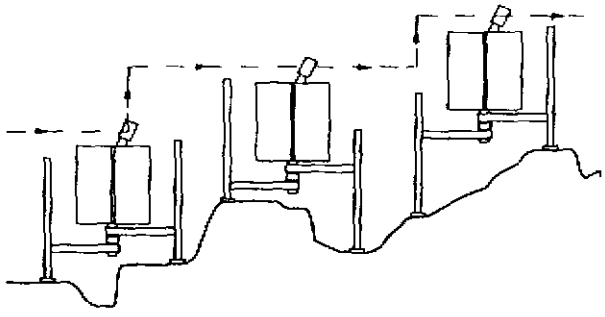


Fig. 3: Level Body Motion

As the body progresses, there is a point at which the rearmost leg must be moved ahead. The act of lifting a leg, moving it ahead, and replacing it on the terrain is termed leg "recovery". Ambler leg recovery is quite unique; after a foot is lifted, the extensional link retracts and the rotational link spins to pass the vertical link between the leg stacks and through the body such that the foot can be placed *ahead* of other supporting feet. During propulsion, supporting legs move rearward relative to the body. Therefore, after every six leg recoveries each leg has completed a full revolution about its respective body shaft. This leg motion, termed circulation, is shown in Fig. 4 where a sequence of six leg recoveries and subsequent body propel motions are shown from left to right across the page. As the walker moves forward the bold leg completes a full counter-clockwise revolution about the left side body shaft. During the same period, it can be seen that all other legs have also circulated to their original positions. Circulation is unprecedented in existing walking mechanisms and in the animal kingdom.

At times the Ambler does not utilize circulation as described. Tight turns require legs on the inside of the turn to recover from front to back (rear circulation), while the outer legs continue to circulate forward (front circulation). For lateral moves, the Ambler uses a traditional insect style "ratcheting" gait in which legs do not pass through the body during recovery.

Elevation maps are constructed from laser range sensor data. Based on the maps, a gait planner determines body and leg trajectories to move the walker toward a goal while maintaining vehicle safety. A graphic simulator ascertains that the planned trajectories are correct. After verificatic trajectories are passed to the robot controller and imposed on the walker. A task-based software architecture connects these various modules and is able to coordinate multiple goals, handle contingencies, and recover from plan failures. Full discussions of the Ambler perception system and gait planning software can be found in (1,2) and (12), respectively. The Ambler's task control architecture is presented in (10).

Models of the Ambler mechanism have been developed to study foot-soil interactions, power consumption, tipover, and foot slippage (8). A comprehensive model incorporates non-conservative foot-soil interactions in a full non-linear dynamic formulation.

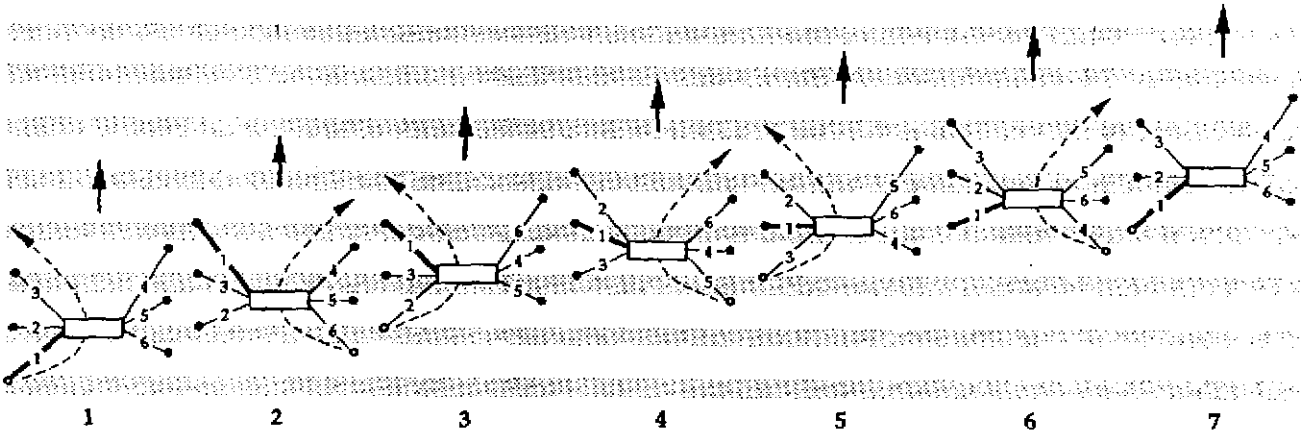


Fig. 4: Circulating Gait. Walker advance is shown in stages from left to right. The dotted curve denotes the path of a recovering leg. Leg #1, the first leg to recover, is shown in bold throughout the sequence to emphasize circulation. After the six steps shown, all legs have completed a full revolution.

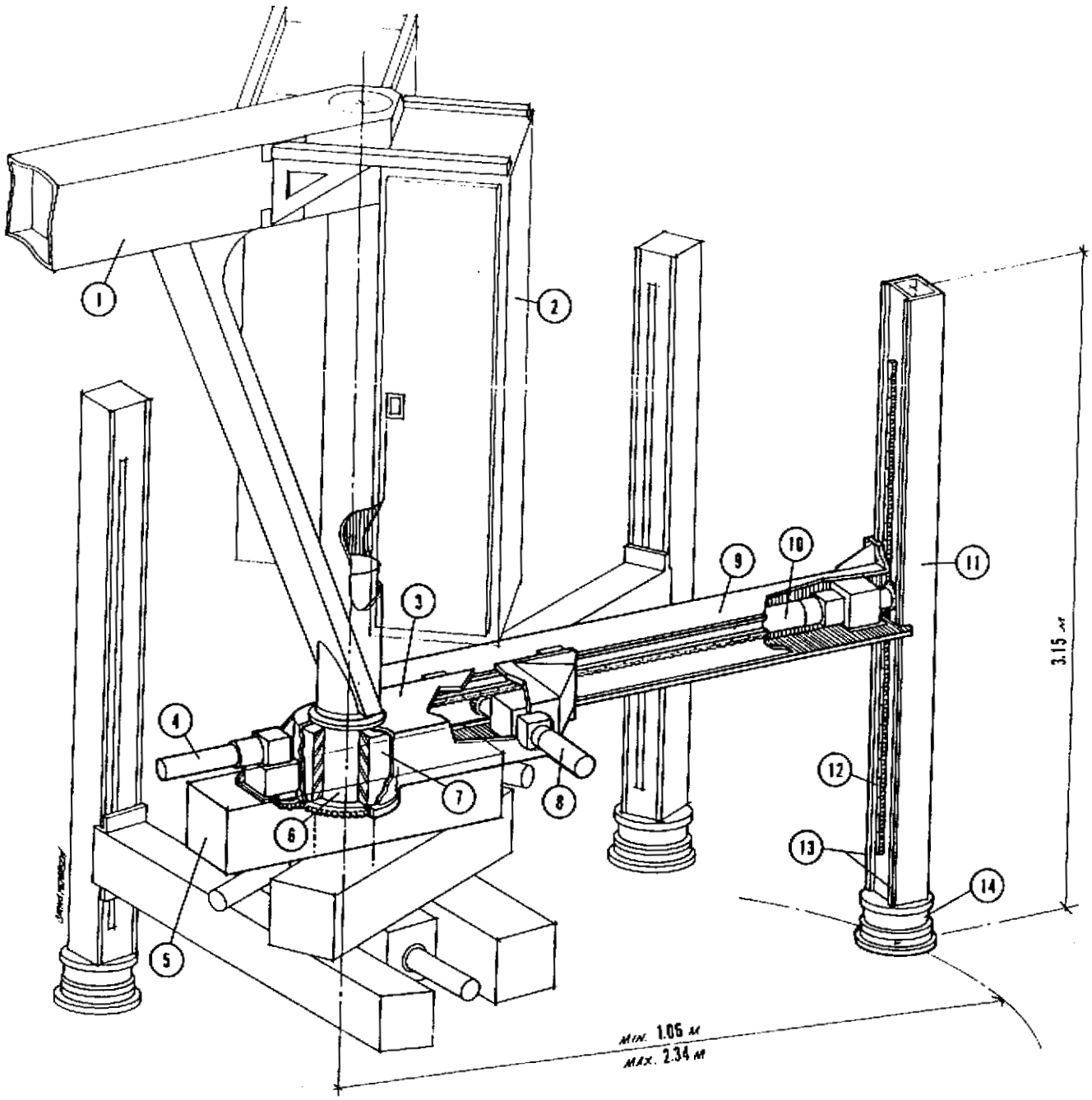


Fig. 5: Ambler Leg and Body Detail

1 – Body structure
2 – Equipment enclosures
3 – Rotational link
4 – Rotational link motor-gearbox
5 – Electronics box
6 – Central shaft
7 – Slipring
8 – Extensional link motor-gearbox
9 – Extensional link
10 – Vertical link motor-gearbox
11 – Vertical link
12 – Rack gear
13 – Linear bearing rails
14 – Force sensor

3.2 Electromechanical Description

Each Ambler leg is identical. Fig. 5 shows three legs, the central shaft about which they rotate, and a section of the body. Components of the uppermost leg have been emphasized. The rotational link gearbox pinion engages a large spur gear affixed to the central shaft. The prismatic links (extensional and vertical) are rack and pinion driven and slide on linear bearings. Each of the three motor-gearbox units include a permanent magnet DC motor, incremental encoder, high efficiency spur gearbox, fail-safe load holding brake, and absolute encoder. A six-axis force/torque sensor mounted to the base of each vertical link provides a full state of force acting on the foot pad.

A multiple-ring slipring commutates power and signals from each leg to the body. Custom digital and analog multiplexors reduce the number of individual rings in the slipring. On each leg, an electronics box mounted to the rotational link houses the multiplexing hardware, motor amplifiers, and brake relays that operate the leg.

The Ambler controller is a combination of single-board computers, motion control boards, and input/output boards on a VME bus. A common real time operating system synchronizes input/output and motion control. Digital boards route a variety of signals including brake control and force sensor control. Analog-to-digital converters read signals from the force sensors, absolute encoders, and inclinometers. A safety circuit monitors all walker motions and immobilizes the robot in response to a variety of sensed unsafe conditions.

A tether connected to the body supplies power as well as linking the robot controller, which currently resides off-board. Eventually, the controller, power generation equipment, and telemetry equipment will be housed in the body, and the tether eliminated.

4. AMBLER CONFIGURATION ADVANTAGES

4.1 Circulating Gait

A traditional rough terrain gait is follow-the-leader: The computer (or human operator) selects front footholds and trailing feet step next to or in the vacated footprints of leading feet (11). There are several reasons why the Ambler's circulating gait should enable it to succeed in terrains that would frustrate or impede walkers with traditional gaits. First, since circulation places recovering feet ahead of supporting feet, the total number of foot placements on the terrain is greatly reduced. A circulating gait requires three to four times fewer foot placements than traditional terrain adaptive gaits (e.g., follow-the-leader). Foothold selection and leg recovery planning difficulty is a function of terrain roughness--as roughness increases, fewer foot placements should be highly advantageous. Furthermore, since foot placements are likely jolt the vehicle's terrain modelling sensors, fewer placements should improve model accuracy.

A second advantage of circulation is that large foot size does not limit rough terrain mobility. For instance, consider a rear-propogating follow-the-leader gait in which trailing feet are placed adjacent to leading feet: Footholds must be large enough to accommodate two feet. Since circulation does not place feet in proximity to each other, footholds need not be oversize. The result is that foothold planning in rough terrain is less constrained and more likely to succeed.

4.2 Decoupled Body Support and Propulsion

Several existing terrain adaptive walkers such as the MELWALK Mark-III (6) and Titan III (3) have leg geometries that decouple body support and propulsion. That is, leg actuation for support is independent from actuation for propulsion and no kinematic coupling exists between the two. Decoupling leads to advantages in efficiency, planning, control, and sensor stabilization. The Ambler's vertical links support and level the body over terrain, and its horizontal links propel the body. As the Ambler moves over terrain the body remains level and is only raised or lowered to pass large terrain features thus minimizing energy expenditure. Once a desired vertical elevation has been selected, body motion planning degenerates to position and heading in the horizontal plane. Propulsion of the level body requires only a determinate subset (three) of the horizontal actuators.

Body support and propulsion decoupling is also advantageous during the development of planning and control software. The Ambler's decoupled configuration permits the vertical links to be locked while propulsion control is developed and alternately, the planar links can be locked while vertical control (e.g., leveling control) is developed. This mode of decoupled development has enabled rapid testing and demonstration of Ambler's basic walking capabilities.

A critical task for an autonomous explorer is mapping its surrounding terrain. The resulting terrain map is the basis for planning and safeguard, so its accuracy directly governs the capability and reliability of the vehicle. Since the perception sensor field of view does not include terrain under or to the immediate sides of the walker and because multiple perspectives of the same scene are required to resolve unknown and occluded areas, registration of multiple images is desirable to build a sufficient map of the local terrain. Maintaining the sensor on a smooth, level trajectory is very advantageous to the speed and accuracy of image registration and correlation.

4.3 Minimal Lower Leg Motion During Propulsion

As the Ambler's body is propelled, the vertical links rotate on the feet without sweeping any volume–especially important in rough terrain where the lower leg could collide with obstacles and possibly become entrapped. This feature of the orthogonal leg should lessen the search for footholds in rough terrain as feet can be placed very near to terrain obstacles (e.g., steps or ledges). Alternately, the shank (lowest link) of a pantograph style leg pivots on the foot and sweeps a volume during propulsion. Allowance for this additional volume must be made during foothold selection, and limits terrain footholds to those with sufficient surrounding clear space.

4.4 Stability and Redundancy

Reliable locomotion is essential for an autonomous mobile robot which will operate on a distant planet. To this end, the Ambler has excellent stability features. First, by maintaining the body in a level configuration, foot forces can be proportioned as desired (within bounds). For instance, a disproportionate amount of the vehicle weight could be placed on the leading legs when hill climbing so that a terrain failure would result in a stumble into the hill instead of a rearward tumble.

Second, the stacked orthogonal leg configuration permits five-legged crawl gaits (i.e., only one leg in recovery at any time) that keep the walker stable even after the failure of any one supporting leg (7). Finally, since legs can operate in each other's zones, the robot can continue statically stable operation after the failure of any two legs. Impaired modes (i.e., four or five operational legs) would probably not use a circulating gait.

5. EXPERIMENTATION

The Ambler (Fig. 6) is currently undergoing walking experimentation on flat, compliant terrain. Development work continues in the areas of perception, gait planning, foothold selection, and mechanism controller. Circulating gait algorithms are being developed that allow general trajectories with curves of any radius. As perception and planning ability matures, terrain roughness will be increased.

After rough terrain capability has been demonstrated indoors, power, computing, and telemetry will be moved on-board the Ambler so that autonomous walking can be demonstrated in rough outdoor terrains. Missions of extreme capability and endurance are envisioned to quantify the walker's ability. Sensors and tools for sampling will be mounted on the Ambler, so that ultimate terrestrial missions can combine long-range walking and sampling tasks.

Mechanism modelling of the Ambler continues to provide simulated time histories and insight about walker performance in situations that are either too difficult or time consuming to test directly on the prototype walker. Modelling and simulation results (e.g., approaches to power efficient walking) are continuously integrated into the Ambler control system.

Fig. 6: Ambler on Flat Terrain

6. CONCLUSIONS

Exploration of rough terrain without continual oversight and input from a human operator has yet to be demonstrated by a mobile robot. The Ambler walker configuration holds the promise of performing successful rough terrain exploration because of its unique amenabilities to sensing, planning, and control–the major elements of autonomy:

- Its circulating gait dramatically reduces the number of foot contacts on terrain thus reducing planning constraints and improving mobility.
- Its decoupled leg geometry and level body motion provide simplifications throughout the system architecture from body motion planning to smooth transport of terrain sensors.
- Its orthogonal leg is ideal for rough terrain as the vertical links sweep no volume during propulsion and can be placed close to obstacles.

The Ambler will be an important benchmark for walker mobility, reliability, and efficiency in rough terrain. Objectives include; autonomous traversal and sampling of terrain more extreme than other walkers have attempted, travel rates and payload that are significant relative to available power, and a mandate for operational modes that never compromise safety.

REFERENCES

(1) J. Bares, M. Hebert, T. Kanade, E. Krotkov, T. Mitchell, R. Simmons, and W. Whittaker. "Ambler: An Autonomous Rover for Planetary Exploration." IEEE Computer, pp. 18-26: June 1989.

(2) M. Hebert, E. Krotkov, and T. Kanade. "A Perception System for a Planetary Explorer." Proc. IEEE Conf. on Decision and Control, Tampa, Florida: December 1989.

(3) S. Hirose, T. Masui, H. Kikuchi, Y. Fukuda, and Y. Umetani. "Titan III: A Quadruped Walking Vehicle." Proc. 2nd Int'l Symp. of Robotics Research, Kyoto, Japan, pp. 247-253: August 1984.

(4) Y. Ishino, T. Naruse, T. Sawano, and N. Honma. " Walking Robot for Underwater Construction." Proc. ICAR, pp. 107-114: 1983.

(5) M. Iwasaki, J. Akizono, H. Takahashi, T. Umetani, T. Nemoto, O. Asakura, and K. Asayama. "Development on Aquatic Walking Robot for Underwater Inspection." Proc. ASME USA-Japan Symposium on Flexible Automation, Minneapolis, MI, pp. 659-664: 1988.

(6) M. Kaneko, M. Abe, and K. Tanie. "Study on Walking Machines with Decoupled Freedoms." J. Robotics and Mechanics, Vol. 1, pp. 21-28: 1989.

(7) S. Mahalingam, and W. Whittaker. "Terrain Adaptive Gaits for Walkers with Completely Overlapping Work Spaces." Proc. Robots 13, Gaithersburg, Maryland, pp. 1-14: May 1989.

(8) D. Manko, "A General Model of Legged Locomotion on Natural Terrain." Ph.D. Dissertation, Carnegie Mellon University, Dept. of Civil Eng.: 1990.

(9) M. Russell. "ODEX I: The First Functionoid." Robotics Age, Vol. 5, No. 5, pp. 12-18: September/October 1983.

(10) R. Simmons, and T. M. Mitchell, "A Task Control Architecture for Mobile Robots." Proc. AAAI Spring Symposium, Stanford, California: March 1989.

(11) S. Song, K. Waldron. "Machines that Walk: The Adaptive Suspension Vehicle." MIT Press, Cambridge, MA: 1988.

(12) H. Thomas, C. Thorpe, and D. Wettergreen. "Planning Strategies for the Ambler Walking Robot." to appear in: Proc. IEEE Int'l Conf. on Systems Engineering, Pittsburgh, Pennsylvania: August 1990.

IEEE International Workshop on Intelligent Robots and Systems
IROS '90

Single Leg Walking with Integrated Perception, Planning, and Control

Eric Krotkov, Reid Simmons, and Charles Thorpe
School of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213

Abstract

We describe an integrated system capable of walking over rugged terrain using a single leg suspended below a carriage that rolls along rails. To walk, the system uses a laser scanner to find a foothold, positions the leg above the foothold, contacts the terrain with the foot, and applies force enough to advance the carriage along the rails. Walking both forward and backward, the system has traversed hundreds of meters of rugged terrain including obstacles too tall to step over, trenches too deep to step in, closely spaced rocks, and sand hills. The implemented system consists of a number of task-specific processes (two for planning, two for perception, one for real-time control) and a central control process that directs the flow of communication between processes. Implementing this integrated system is a significant step toward the goal of the CMU Planetary Rover project: to prototype an autonomous six-legged robot for planetary exploration.

1 Introduction

The goal of the CMU Planetary Rover project is to prototype an autonomous mobile robot for planetary exploration. The design is a six-legged walking robot with orthogonal legs and an overlapping gait [Bares, this proceedings]. To successfully walk over rugged terrain, the rover must combine perception, planning, and real-time control in an integrated system.

Recent research toward such an integrated system has concentrated on the task of single leg walking as a special case of six-legged walking. What distinguishes our work is the simplicity of the walking mechanism and the completeness and comprehensiveness of the controlling system. Other researchers use a single leg to isolate and study fundamental issues in balance and dynamics [9, 11]. Our reasons for using a simple mechanical system—it is stable both statically and dynamically—include testing algorithms with relative safety and ease, and coordinating design and development so that results from walking experiments influence more quickly the evolution of design of the six-legged walker. Previous efforts to create a coherent robotic walking system from component research results have generated significant advances, for example as reported in [5, 6, 7, 10] and many others. One of the reasons that these efforts have not proven entirely effective is the difficulty involved in developing each of the subsystems (e.g., locomotion, perception, planning, control); thus, each research effort has concentrated on a proper subset of the issues.

This paper describes the comprehensive system that we have implemented, and presents results from single leg walking experiments. We refer readers interested in the objectives and accomplishments of the overall project to [12].

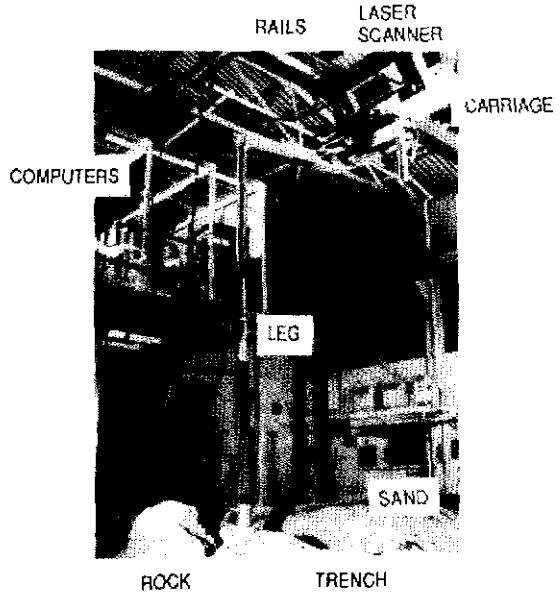


Figure 1: Single Leg Testbed

2 Single Leg Testbed

The single leg testbed contains over 40 tons of sand and a variety of obstacles in an 11×6m "sandbox" made from I-beams. It also includes a robot leg, sensors, and electronics (Fig. 1).

The robot leg is a prototype design not currently used on the six-legged Ambler (because integrated walking experiments revealed problems early). The leg has three joints: a revolute shoulder, a revolute elbow, and a prismatic vertical axis. Its horizontal length is 2.5m and vertical travel is 1.5m. In the shoulder and elbow axes, brushless DC servo motors couple to the joint axes by an 80:1 harmonic drive speed reducer and a 3:1 bevel gear. In the vertical axis, a brushless DC servo motor drives a 12:1 speed reducer and a lead screw.

The leg hangs from a carriage (or body) that rolls along rails. The leg "walks" by planting the foot on the ground and actuating the shoulder and elbow motors to push or pull the carriage.

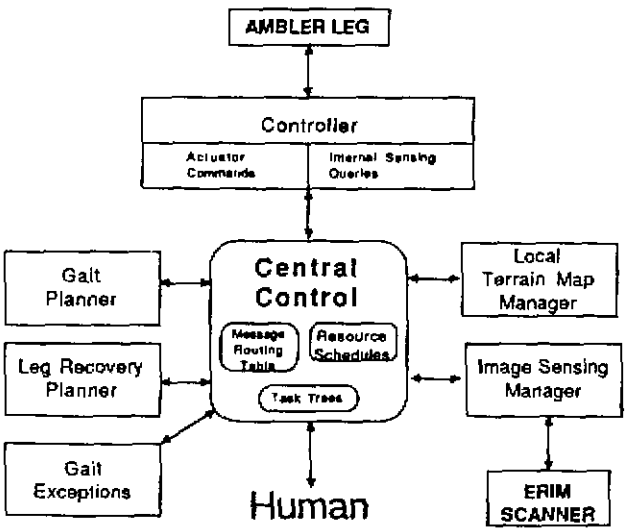


Figure 2: Modules of the Single Leg Walking System

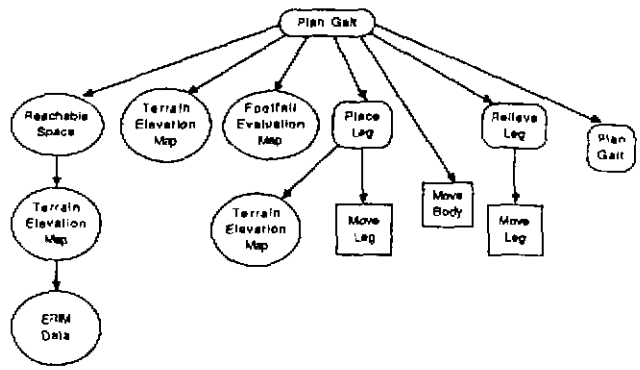


Figure 3: Task Tree
Bubbles represent query messages, rectangles represent command messages, and rounded rectangles represent goal messages.

Although the rails are I-beams, they exhibit significant deflection due to their length and the loads applied. This deflection provides compliance for the system (which is good), but changes as a function of the leg position (which is bad, because it makes accurate leg placement difficult).

Sensors on the leg include motor shaft encoders, limit switches, and a six-axis force/torque sensor mounted near the foot. Other sensors include a scanning laser rangefinder to sense the terrain in the sandbox, a potentiometer that measures the distance travelled by the carriage, and two inclinometers to measure rotation of the carriage with respect to gravity.

A control room houses three workstations connected by Ethernet, hardware to control the laser scanner, and a VME cage containing a real-time control system with its associated hardware (68020 single board computer, Ethernet controller, A/D converter, two 80186 motion control cards, and an interface card to connect them to the motor amplifiers).

3 Task Control Architecture

Simmons et al. designed the Task Control Architecture (TCA) to integrate sub-systems developed by different researchers into a complete robotic system [8]. TCA provides mechanisms to support message passing between distributed processes, hierarchical planning, plan execution, monitoring the environment, and exception handling. A system built using TCA consists of a number of task-specific processes, called *modules*, and a general *central control* process that directs the flow of communication between modules. The single leg walking system consists of six modules plus the central control (Fig. 2).

A prominent aspect of TCA is centralized control. Although researchers have recently advocated decentralized control for mobile robots, e.g [3], the TCA designers believe that centralized control has many advantages for supporting the above capabilities. First, it can more easily control multiple tasks by synchronizing them, allocating resources, and determining which tasks have priority. Second, centralized control makes the system more understandable and easier to modify. Since there is a single point through which all communication flows, one can easily monitor and analyze the communication. Finally, we have not found centralized control to be a system bottleneck in our applications; TCA can process a message in approximately 80msec, faster than either the perception or control systems operate.

Modules can connect with the central process in any order and at any time. If a module crashes, it can be restarted and reconnected without bringing down the rest of the system. Modules interact with TCA by calling utility functions, passing as arguments standard C or LISP data structures. When modules connect to the central control, they indicate which messages they can handle by registering the message name, a handler procedure, and the data format of the message. TCA also contains facilities for displaying arbitrary data structures; we use this to log all message traffic.

In TCA, planning and executing a task occurs by modules sending a series of messages to one another. For the single leg walker, after all modules have connected to central and registered their messages and handlers, a message is sent to the gait planner instructing it to begin planning. To plan and execute a complete step involves sending about 25 messages.

These messages are of various types—each message class has a different semantics and different effects. *Query* messages obtain information about the external or internal environment. *Goal* messages provide a mechanism for hierarchical planning. When a module issues a goal message, TCA creates a node and adds it as a child of the node associated with the handler that issued the message. These nodes form a hierarchical *task tree* that TCA uses to schedule planning and execution of tasks (Fig. 3). *Command* messages are requests for some action to be performed. Like goal messages, TCA adds them to the task tree; typically, they form the leaf nodes of the tree.

In addition to specifying parent/children relationships in the task tree, TCA provides mechanisms for temporally constraining the relationships between nodes in the tree. Essentially, the task trees plus the temporal constraints form TCA's representation of

plans. For example, one can specify that command A must be executed before command B is started, or that goal C cannot be planned until command B has finished. TCA maintains separate constraints for the planning and achievement of tasks—thus, one could specify that the robot should go to a sample site and then acquire a sample, but that it should plan how (and if) it can acquire the sample before planning how to navigate to the site.

Unlike query messages, goal and command messages are non-blocking, i.e., a goal or command message has not necessarily been handled by the time control returns to the module issuing the message. This asynchronous control makes the overall system more reactive since the central process controls when to schedule tasks and when to preempt them. The non-blocking nature of goal and command messages also makes it easy to do planning in advance of execution. The planning modules merely send messages that create task trees, and TCA ensures that the tasks will be executed at the appropriate times.

In addition to separating planning and execution, TCA uses a separate mechanism to perform exception handling. Goal and command messages, when they detect plan-time and execution-time failures, respectively, issue exception messages. The central control suspends the current task and routes the exception to the appropriate user-defined handler. There, the exception handler can analyze the failing task tree and decide how to manipulate it to recover from the error, for instance, by killing part of the task tree, or by adding new nodes to the tree to patch the plan.

4 Real-Time Control

The control software runs on the real-time system under the vxWorks (TM) operating system, and communicates to the rest of the system through TCA. The controller performs three tasks: it executes leg and body movements and reports their positions; it communicates with the user (either a person or a process); and it handles asynchronous interrupts generated by the motion control cards. This section discusses only the first of these tasks.

Given a series of points in joint space (way-points), the controller actuates the leg motors so that the leg passes through each. It computes the time required for the slowest joint to move between successive way-points, and then scales the speeds of the other joints so that all arrive simultaneously at each way-point. To smooth the motion, the controller links way-points with constant velocity segments which in turn it connects by constant acceleration segments. If the user specifies the last path segment to be in transition mode, then the controller places the foot on the ground and loads it up to a specified force.

Once it achieves the desired load on the leg, the controller moves the body by actuating the shoulder and elbow joints to move at given velocities, not to given positions. It performs this at about 60Hz, which differs sufficiently from the natural frequency of the system to preclude resonance. The controller computes the joint velocities by applying the inverse Jacobian to the Cartesian body velocity, which is a clipped, linear function of the error between the current and commanded body positions. Due to non-linearities of the system, this causes overshoot of the joints from their nominal position given a perfectly linear system. This overshoot takes the form of stored strain energy. The controller dissipates the strain before unloading the leg, otherwise the foot could drag across the ground, possibly hitting an obstacle. If at any time the forces exerted on the foot decrease rapidly, indicating that the foot has lost contact with the ground, the controller halts.

5 Perception

The perception system consists of two major modules: the Imaging Sensor Manager (ISM), which senses the environment with a scanning laser rangefinder; and the Local Terrain Map Manager (LTM Manager), which constructs elevation maps from the rangefinder data. Readers will find a higher level account of the perception system in [1] and a lower level account in [4].

The ISM operates the imaging sensors, including initialization, status determination, data acquisition, calibration, aiming, and other operations. The ISM has been implemented and tested for the Erim and Perceptron scanning laser rangefinders. These sensors may be real (i.e., they acquire data in real-time from the physical sensor) or virtual (i.e., they acquire data from storage, not directly from the sensor). We have found virtual devices and virtual images to be useful for developing and testing code without hardware.

The LTM Manager constructs and maintains a local terrain map (LTM)[1] for locomotion guidance, short-range navigation, and sampling operations. An LTM describes the environment in the immediate vicinity of the Ambler, and may extend up to tens of meters on a side. An LTM is not, strictly speaking, a single map; in practice, it is a registered collection of maps, whose descriptions of the environment include geometric characteristics and material properties of the terrain. We have organized the software into three major submodules: one that builds the LTM, one that merges LTMs, and one that focuses attention on parts of the LTM that are closer to the vehicle (not described here).

The LTM Builder constructs an LTM from a single frame of sensor data by transforming the raw sensor observations into a structured description of the terrain in the local vicinity of the vehicle. The implementation uses the Locus Method [4] to transform the input raw range images into an output elevation map. In addition, the LTM Builder computes the uncertainty of the estimated elevations, analyzes elevation map patches as footholds, and estimates the mean slope over elevation map patches.

The LTM Merger maintains the LTM to reflect the information contained in a sequence of maps constructed by the LTM Builder. The implementation of the LTM Merger accepts as input the LTM L_k^0 constructed from range images $I_0, I_1, \ldots, I_k$, and the LTM L_{k+1}^{k+1} constructed from range image I_{k+1}. It generates as ouput the LTM L_{k+1}^0 by replacing overlapping elevation measurements with the maximum likelihood estimate of the elevation. The merging operation is necessary 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 range image can not possibly see obstacles either below or behind the vehicle. These obstacles pose real threats to the recovering leg, which must follow a trajectory that avoids collisions with them. Thus, the merging operation is necessary to create an LTM that

[1] To eliminate any possible confusion about the terminology, we mean to distinguish the LTM, which is a data structure, from the LTM Manager, which is a process.

provides a wider coverage of the terrain than is possible with a single frame of data.

6 Planning

The planning problems for single leg walking include deciding where to place the leg, how to move it there, and how far to move the carriage at each step. The planning system consists of two modules: the gait planner chooses footholds and body advances, and the leg-recovery planner identifies trajectories from the current leg position to the planned foothold.

The gait planner computes *cost maps* that indicate the "goodness" of each potential foothold on a 10cm grid. It assigns costs based on the following constraints: 1) flat terrain is preferable both for stability and for providing traction in moving the body; 2) the carriage can advance farther from some footholds than from others; 3) leg configurations in which horizontal links obstruct the scanner field of view are undesireable; 4) the leg can not reach areas outside its kinematic limits or ones surrounded by high obstacles (including other legs, for the six-leg case); 5) the leg can not reach terrain that is too high or too low (recall that the body height is fixed). The gait planner combines the cost maps using a weighted sum. It selects as the foothold the grid point with the lowest cost in the composite cost map. It plans the body move that is the minimum of 1) the largest advance possible from the chosen foothold, and 2) a user-defined threshold.[2]

Advantages of this constraint-based approach are that the planner does not have to commit *a priori* to which constraint is most important, and it is easy to add new constraints as relevant ones are identified. Although this approach could result in high computational costs, in practice the planner is fast relative to other computations.

While the gait planner decides where to set the foot, the leg-recovery planner determines the trajectory to that position without hitting obstacles. The leg-recovery planner uses a novel algorithm that finds time and power efficient moves through three-dimensional space while searching only a two-dimensional space, thus considerably increasing the efficiency of the planning.

The planner creates a configuration search space for the elbow and shoulder joints. It divides the space into a discrete grid approximately 0.1 radian wide, and fills the grid with obstacles. It grows terrain obstacles and other legs (for the six leg case) by the radius of the foot plus an uncertainty factor. The planner then searches this space using the A* algorithm for the minimum cost path (weighting power and time by a user-specified ratio) to the goal, either by going around or over obstacles. It computes the power consumed to reach a grid cell from an adjacent cell as the sum of the power needed to move the elbow and shoulder joints to get to the cell, plus the power needed to raise the leg above the elevation associated with that cell. It computes the time required to get to a cell by keeping track of 1) all possible paths that the leg can take in reaching a particular grid cell, and 2) the maximum and minimum heights that the leg can reach at any particular cell, assuming that the leg lifts/lowers at full speed while moving horizontally. At the end of the search, the planner determines the final trajectory by selecting vertical moves that minimize the risk to the machine while maintaining the optimality of the path found.

[2]The threshold cannot exceed the maximum body advance of 3.9m. In order to take more steps per experiment, typically we use 1.5m.

Figure 4: Obstacle Course

The obstacle course is 10m long and consists of a box (right) too tall for the leg to step over, a "steeplechase" arrangement of pylons (center) lying on the ground, two larger obstacles (left and upper center) separated by about 1m, and a dozen or so smaller obstacles.

7 Experiments

After the single leg testbed became operational in May 1989, we performed a series of tests on half a dozen obstacle courses. Fig. 4 shows one of them, and Fig. 5 illustrates a map of it constructed by the perception system. The courses combine obstacles that are too tall to step over, obstacles separated by distances smaller than the diameter of the foot, trenches too deep to step in, and sand hills with a variety of slopes. The criteria for a successful traversal are to reach the goal and to avoid contact with any obstacle.

For each trial, first we activate the integrated system shown in Fig. 2. Then we issue a command to walk forward to the end of the testbed (about 10m). After this, the integrated system is entirely autonomous as it plans and executes the walking cycle of moving the leg and propelling the body along the rails.

The integrated system successfully negotiated all of the obstacle courses. It traversed the course in Fig. 4 seven consecutive times during one afternoon. It traversed comparable courses more than thirty times. In some of the trials, the system also walked backward, using the map built by the perception system while walking forward. This was surprisingly easy; the system could always follow a successful forward traverse by a successful backward traverse.

Not all forward walking trials were successful. Failures include stepping on obstacles, and grazing them with the leg. To diagnose these failures is challenging. To illustrate the difficulty, suppose that the leg strikes an obstacle. What went wrong? The fault could be in any of the subsystems. The perception system may have computed an inaccurate map, incorrectly determining

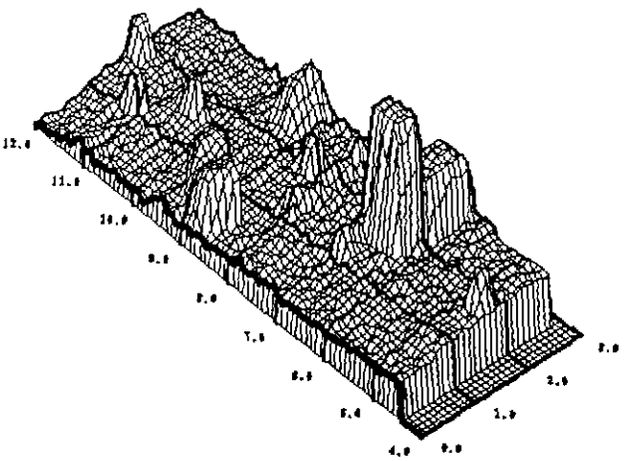


Figure 5: Elevation Map of Obstacle Course
The perception system built this elevation map from five range images acquired at different positions. The map resolution is 10cm, $0 \leq X \leq 3$m, and $4 \leq Y \leq 12$m.

the obstacle location. The planning system may have chosen a poor foothold or an erroneous leg trajectory. The controller may have executed poorly a perfect plan. Or any combination of the former factors could cause the fault.

During the walking trials, we identified the the following combination of factors to be responsible for the largest number of failures: 1) the perception system sometimes underestimates the possible error in the perceived location of an obstacle, 2) the controller does not compensate for the change in leg positioning accuracy as a function of body position,[3] 3) the gait planner, working without models of the above deficiencies, is sometimes more eager to advance to the goal (i.e., plan longer steps) than to steer clear of obstacles (i.e., plan steps that sacrifice body advance for obstacle avoidance).

We find the average walking velocity to be on the order of one meter per minute. Since we have not dedicated much effort to optimizing either hardware or software, this statistic may not be particularly meaningful. One version of the integrated system achieves nearly continuous walking (Fig. 6). It concurrently executes one step while planning the next step(s), exploiting the temporal constraint mechanisms of TCA.

8 Discussion

We have described an integrated system that combines advanced techniques in perception, control, and planning into a comprehensive whole. Experiments show that the system can perform capable and fairly reliable single leg walking on rough terrain.

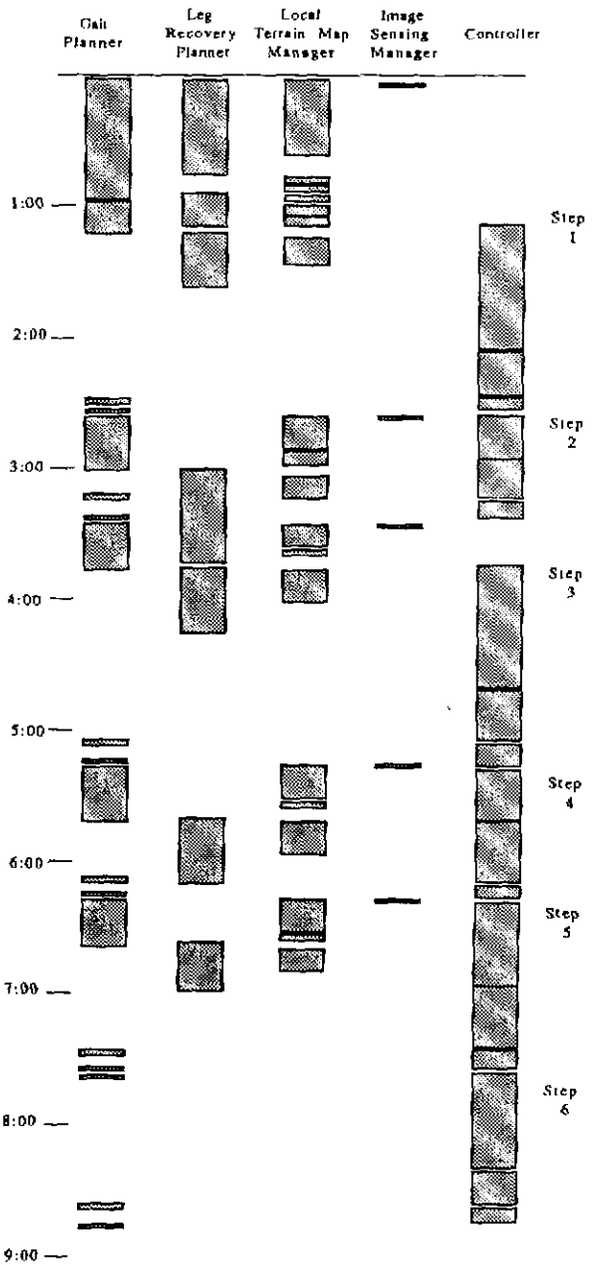


Figure 6: Nearly Continuous Walking
The figure shows when processes are active during one obstacle course traversal using concurrent planning and execution. Numbers in the left column indicate time. The controller is idle between Steps 2 and 3 because the leg recovery planner takes longer than usual to plan a complex trajectory around (not over) the large box shown in Fig. 4.

[3] Due to variable compliance of the rails, the errors in leg positioning grow with the distance along the rails from the body to the closest anchoring wall. Thus, the errors are larger in the middle of the testbed than at the ends.

The major impetus for the single leg walking research program was to gain experience for six leg walking. In that regard, the project is quite successful. We have gained insight into controlling the legged mechanism, calibrating the leg and scanner, planning in the face of uncertainties and conflicting constraints, and coordinating a distributed software system.

Future work will concentrate on applying our experiences to an integrated system for six leg walking. This will require significantly extending the controller and the planners, but only superficial changes to the perception system and the TCA.

One topic that we anticipate will be an issue in the future is calibration of the leg and scanner frames. For the six-legged walker, we may augment our current model-based calibration with a direct, empirical method that, for example, locates the leg in many images and uses a connectionist approach to build an inverse kinematics table [2].

Another topic for future work is better performance in situations that conflate errors in perception, errors in control, and optimism in planning (cf. the cause of the failures cited in Section 7). To better understand situations that cause failures we need more powerful debuggers (possibly graphical). To achieve more reliable performance we must develop robust error recovery mechanisms.

In this paper we have concentrated on the integrated system. We conclude by discussing briefly the process of system integration. Perhaps the most important lesson that we learned is that integration is a contact sport; it cannot succeed without significant "hands on" participation by researchers with a broad range of experience and expertise. While this may be self-evident, it is by no means easy to accomplish.

We have adopted several approaches to facilitate the integration effort. First, we conduct regular weekly meetings to identify the semantics for all interfaces between modules, answering questions about the type and units of information communicated. The message-passing conventions of TCA promote this because they force us to detail the interfaces. Second, we employ Unix manual facilities to document the I/O behavior of a module or message. We find this to be a useful lowest common denominator that all programmers and users can use to advantage and that is not so difficult to maintain. Third, we insist that software meet standards of internal documentation (comments for modules, files, and functions), follow conventions for naming (source files, include files, defines, type definitions, and variables), and obey a variety of other guidelines. We find this to be of great value both in debugging and development. Finally, we standardize our software structure in order to make common code more accessible. This includes enforcing consistent directory structures and naming conventions.

These and other approaches have significantly assisted us in integrating the single leg walking system. We intend to continue these practices as we combine our perception, planning, and control techniques into a comprehensive system for six-legged walking.

Acknowlegements

We would like to acknowledge contributions by all the members of the Planetary Rover project, and thank especially P. Balakumar, L. Chrisman, C. Fedor, M. Hebert, G. Roston, and D. Wettergreen for their assistance in integrating and testing the single leg walker.

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

References

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

[2] J. Barhen, S. Gulati, and M. Zak. Neural Learning of Constrained Nonlinear Transformations. *IEEE Computer*, pages 67–77, June 1989.

[3] R. A. Brooks. A Robust Layered Control System for a Mobile Robot. *IEEE Journal of Robotics and Automation*, RA-2(1):14–23, March 1986.

[4] M. Hebert, E. Krotkov, and T. Kanade. A Perception System for a Planetary Explorer. In *Proc. IEEE Conf. on Decision and Control*, pages 1151–1156, Tampa, Florida, December 1989.

[5] S. Hirose. A Study of Design and Control of a Quadruped Walking Vehicle. *International Journal of Robotics Research*, 3(2):113–133, Summer 1984.

[6] Y. Ishino, T. Naruse, T. Sawano, and N. Honma. Walking Robot for Underwater Construction. In *Proc. Intl. Conf. Advanced Robotics*, pages 107–114, 1983.

[7] C. A. Klein, K. W. Olson, and D. R. Pugh. Use of Force and Attitude Sensors for Locomotion of a Legged Vehicle over Irregular Terrain. *International Journal of Robotics Research*, 2(2):3–13, 1983.

[8] L.-J. Lin, R. Simmons, and C. Fedor. Experience with a Task Control Architecture for Mobile Robots. Technical Report CMU-RI-TR-89-29, Robotics Institute, Carnegie Mellon University, 1989.

[9] M. H. Raibert. *Legged Robots That Balance*. MIT Press, Cambridge, Massachusetts, 1986.

[10] S. Song and K. Waldron. *Machines that Walk: The Adaptive Suspension Vehicle*. MIT Press, Cambridge, Massachusetts, 1988.

[11] M. Sznaier and M. J. Damborg. An Adaptive Controller for a One-Legged Mobile Robot. *IEEE Transactions on Robotics and Automation*, 5(2):253–259, April 1989.

[12] W. Whittaker, T. Kanade, and T. Mitchell. 1989 Year End Report: Autonomous Planetary Rover at Carnegie Mellon. Technical Report CMU-RI-TR-90-4, Carnegie Mellon University, February 1990.

To appear in: Proceedings of the IEEE International Conference on Systems Engineering, August 1990.

Planning Strategies for the Ambler Walking Robot

David Wettergreen, Hans Thomas, and Charles Thorpe

The Robotics Institute, Carnegie Mellon University
Pittsburgh, Pennsylvania 15213

Abstract: Mobile robots that autonomously explore rugged natural terrain must be highly adept at locomotion requiring competent planning strategies in addition to a capable vehicle. We propose and explain a hierarchy of planning strategies for a *walking* robot, the Ambler. The hierarchy decomposes planning into levels of trajectory, gait, and footfall. An abstraction of feasible traversability allows the Ambler's trajectory planner to identify acceptable trajectories by finding paths that guarantee footfalls without specifying exactly which footfalls. Leg and body moves that achieve this trajectory can be generated by the Ambler's gait planner which incorporates pattern constraints and measures of utility to search for the best next move. By combining constraints from the kinematics of the mechanism with constraints from the quality and details of the terrain, the Ambler's footfall planner can select footfalls that insure stability and remain within the tolerances of the gait.

Introduction

Mobile robots that autonomously explore rugged natural terrain must be highly adept at locomotion requiring competent planning strategies in addition to a capable vehicle. We propose and explain a hierarchy of unique planning strategies for a *walking* robot. These strategies decompose rough terrain navigation into levels of resolution that allow novel abstractions and simplifications of the problem. Separate planners in a hierarchy can cooperate effectively if the decomposition is correct. For localized walking, on the scale of fifty meters, planning can be decomposed into levels of trajectories, gaits, and footfalls.

Trajectories link position and orientation objectives in the environment. The trajectories intended here are analogous to the trajectories that connect the knotpoints, or goals, in the configuration space of a standard fixed-base manipulator. An abstraction of "feasible traversability" allows a trajectory planner to identify acceptable trajectories without concern for details at lower levels in the hierarchy by finding paths that guarantee acceptable footfalls without specifying exactly which footfalls.

A *gait* is a sequence of leg and body moves. The usual context of gait studies is the regular, periodic combinations of legs on the ground and leg motions. These different patterns generate trots, canters, gallops and so forth. Walking robots in very rough terrain have a completely different set of constraints. For maximum stability, only one leg is moved at a time. Because of the tight maneuvers and rugged terrain, the sequence of leg movements may not be periodic or fixed but may vary continuously. In this context, gait planning is often a moment-by-moment analysis of which leg should be moved, and approximately where it should be placed. Leg and body moves that achieve a given trajectory can be generated by a gait planner that incorporates pattern constraints to search for the best next move.

Individual *footfalls* are the specific terrain contact points that are derived from the gait pattern. The horse's ability to avoid rocks and fences as it gallops is evidence that the specific foot contact is allowed some variation within the constraints of the gait pattern. By combining constraints from the kinematics of the mechanism with constraints from the quality and details of the terrain, a footfall planner can select footfalls that insure stability and remain within the tolerances of the gait.

The Ambler Walking Robot

Rugged terrain is characterized by featureless landscapes of sand and rock containing obstacles of irregular geometry. Mining, construction and waste disposal sites, and planetary surfaces are examples of rugged terrain. The Martian surface typifies these environments. (Figure 1)

Figure 1 - Surface of Mars from a Viking Lander

Although the problems involved in rugged terrain navigation have been investigated, the vehicles used in these experiments have been primarily wheeled or tracked locomotors. [Rosenblatt88] [Stentz89] Legged vehicles offer advantages in control of stability, isolation from terrain irregularities, power consumption, and rough terrain capability.

The *Ambler*, for autonomous mobile exploratory robot, is a walking robot built at Carnegie Mellon University to traverse rugged terrain with high reliability. (Figure 2) [Bares90] The Ambler is a unique mechanism with six legs each of which consists of two links in the horizontal plane, one rotational and one extensional, and one extensional link in the orthogonal, vertical plane. The rotational links are stacked around two central shafts with three legs on each and are able to rotate continuously. The horizontal links permit planar motion of the leg and the vertical extensional link provides motion of the foot down to ground contact.

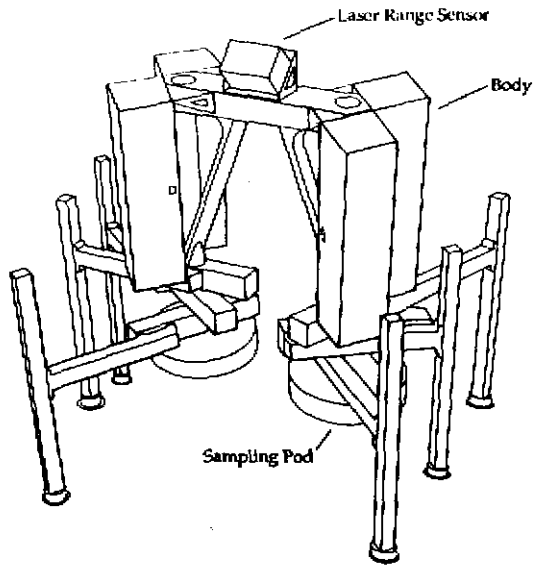


Figure 2 - Ambler Walking Robot

On-board sensing consists of a scanning laser range-finder and foot-mounted force-torque sensors. The range finder is used to generate depth maps of the terrain. [Kweon89] At close range the depth maps are of high resolution suitable for selecting individual footfalls, while at long range they provide adequate information to determine the traversability of distant terrain. The force-torque sensors provide support information about the terrain and the stability of the current *stance*, the static arrangement of the legs and body. Although the specific position of the legs changes the overall height, width and length of the Ambler, a typical stance is 5 meters tall, 4.5 meters wide and 3.5 meters long.

The orthogonal leg design of the Ambler decouples horizontal and vertical motions for energy and planning efficiency. Each vertical link adjusts to the terrain roughness so that the Ambler remains constantly level providing a stable platform for sensing and sampling operations.

The Ambler walks by lifting a leg vertically, swinging it in the horizontal plane, extending it down to terrain contact, and then gliding the body forward at level elevation by coordinated actuation of the joints in the supporting legs. The Ambler can perform any of the gaits associated with hexapods including the alternating tripod, in which three legs move while three support, and crab walking which is an alternating tripod in the lateral direction. Follow-the-leader, ratchet or wave gaits, where some legs move to new locations while others adjust into available locations — in the manner of centipede — are also possible for the Ambler.

The Ambler's primary mode of walking is a *circulating* gait. This gait, which has no natural counterpart, is performed by lifting a rear trailing leg, passing it through the body cavity, and placing it out in front of a leading leg. When this circulation of the rear legs to the front is repeated six times, all six legs will have made a complete circulation of their stack axis. The circulating gait can reduce the number of footfalls per equivalent body advance to less than that of any naturally occurring gait. By circulating one stack forward and one stack back (retrograde) the Ambler can turn in place. Through proper selection of gait and foot placement, the Ambler can follow an arc of any radius (from zero for a point turn, to infinity for a straight line).

Trajectories

Trajectories define the desired vehicle path through ten to twenty meters of terrain, limited by the sensor field of view. The Ambler needs a trajectory planner that can contend with occluded regions, limited resolution, and the computational explosion of potential paths, and still find acceptable trajectories to follow. It must enable the Ambler to avoid vehicle-sized obstacles and otherwise impassible regions. The challenge of trajectory planning for a walking vehicle is to find an abstraction for traversability that is tractable to calculate but does not simplify the problem to the point that only the most ideal cases can be addressed.

It is possible, but undesirable, to search for a trajectory by enumerating and testing each potential footfall and the actuating motions. This would have the advantage of finding all possible trajectories, but is unrealistic for actual operation. The difficulty is the overwhelming amount of data to consider. Any exhaustive approach is infeasible in all but the simplest cases.

The opposite approach to exhaustive search is to select trajectories through "smooth" or "easy" terrain, as determined by a simple terrain-smoothness operation, and gamble that, as the robot approaches each point along the path, suitable footfalls will be found. Such an abstraction operates quickly but may generate infeasible trajectories, or overlook feasible trajectories, because its abstraction of terrain over-generalizes.

The Ambler uses a strategy that improves on both of these approaches, by developing a terrain representation that reduces the computational and data representation complexity, without losing the ability to distinguish feasible from infeasible trajectories. The result of this planner is a path with high probability of successful execution without the expense of individual foot placement enumeration.

The most difficult part of trajectory planning for the Ambler is determining where, on the terrain, the vehicle can feasibly stand. Models of the robot's kinematic capabilities, maneuverability, and support constraints provide reasonable measures that can be used to evaluate terrain for traversability. The important considerations for stability are the body height and feasible footfall locations. Minimum body height is limited by the highest point below the vehicle. Given the minimum body height, a set of feasible footfalls are determined by maximum vertical leg extension and feasible horizontal leg placements. The strategy for a given x and y position is to:

- Calculate the minimum Ambler height so the body or sweeping legs just contacts the highest point of nearby terrain.
- Find all reachable footfalls. Footfalls must be acceptable in vertical extension as well as horizontal reach.
- Examine the set of reachable terrain points to identify a set of five feasible footfalls that provide stable support. (Figure 3) If a set exists, then this terrain patch will support the vehicle; if not, it is impassible.

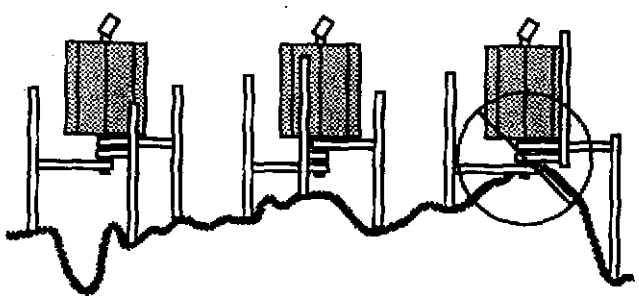


Figure 3 - Low traversable, high traversable and impassible terrains.

In moderate terrain, it is possible to look at larger areas of terrain. If the maximum difference in elevation over an area of terrain is less than the leg *stroke*, the range of vertical leg extensions, then all points in that area are feasible footfalls, and the terrain is automatically traversable. This simplification is particularly useful in moderate terrain characterized by small surface slope and sparse obstacles.

In the opposite extreme there are pathological cases in which the Ambler can stand in various locations in the terrain but cannot move from one to the next. This generally occurs when the number of changes in leg placement from one stance to the next is high. Such failure cases indicate two things. First, that the terrain is extremely complex — subsequent failure by lower level planners will indicate that it is impassible. Second, the trajectory planner is not operating at fine enough resolution in position and/or orientation to detect impassible intermediate terrains. Failure indicates that the trajectory planning should be repeated at finer resolution. There is a clear relationship between the complexity of the terrain and the resolution of trajectory planning required.

Given a traversable passage, a specific path has to be found for the Ambler to follow. The Ambler lacks the non-holonomic constraints found in most conventional wheeled vehicles. It possess the ability to vary its foot placement and perform omnidirectional body motion. Unlike a car, it is not constrained to navigate along arcs greater than some minimum radius — turns of zero radius are possible. Although this is attractive from the standpoint of agility, it complicates the planning process. A sequence of point-turn to face the goal, move in a straight line to the goal position, and point-turn to the goal orientation, can connect any positional objectives by the shortest path but not necessarily by the least number of footfalls. In many cases an arc is a more efficient trajectory between goals. The planning strategy must identify the best path within the traversable area to attain positional goals and we are currently exploring ways of selecting such a path. We will be developing measures of the relative expense in energy, time, and stability of various arcs and transitions, and using them to search for the best path. A method developed on the Carnegie Mellon Navlab, to build a configuration space using similar metrics, holds promise for the Ambler as well. [Stentz89] The actual selection of a trajectory is then a search through feasible positions and orientations in the configurations from the start point to the goal.

Gaits

When the vehicle trajectory has been designated, a gait that executes this trajectory efficiently must be planned. The trajectory may be complicated. It may include turns, reversals, and lateral motions. Much of the research performed to date on gaits concerned itself with either analyzing the patterns underlying the gait or the realization of a particular gait on a physical mechanism. Although this has succeeded in identifying the major elements which characterize a gait (such as the *pitch* which is the spacing between legs along the direction of motion and the *tread* which is the distance from the direction of motion to the foot) as well as several dominant gait patterns (like alternating tripod and follow-the-leader) a method for generating a gait from basic characteristics has not been forthcoming.

Walking can be generated by combining several independently-operating behaviors as demonstrated on a recent walking insect robot. [Brooks90] The walking insect featured a rear-propogated wave gait implemented as a finite state machine, with legs adapting to minimize the body tilt. This algorithm is similar to that employed by the Adaptive Suspension Vehicle (ASV) and the MELWALK III vehicle. [Song88] [Kaneko89] The walking insect was also able to learn to walk by converging on regular pattern of motion. This was particularly interesting because it demonstrated the

fundamental nature of the alternating tripod gait and the notion that gait is essentially a pattern generation problem.

The ASV demonstrated the feasibility of rugged terrain locomotion with a legged vehicle, using a human "planner". [Lee88] Various gaits were developed for traversing moderately difficult terrain. In order to overcome large obstacles, however, pre-programmed walking modes for specific classes of obstacles (ditches, walls, hills) had to be initiated by the operator. More desirable is a mode-less form of walking in which the direction of travel generates the gait, rather than by selection from a discrete set of options. The ideal mode-less walking is a smooth glide over terrain with no rough transitions.

The significant gait characteristics necessary to achieve arbitrary motion are the length and direction of the stride, the width of its stance, and the sequence of leg recoveries. A global search for appropriate values quickly leads to a combinatorially explosive situation, even for short vehicle motion. Such global optimization is unnecessary — gait is a local phenomenon. The choice of a footfall or body move at one point in time rarely has any relevance to footfalls or body moves far in the future. It must be opportunistic to take advantage of the vehicle's current configuration, desired path, and terrain. The primary constraints governing gaits are also local in nature, such as maintaining stability and maintaining advance. In order to plan a gait that satisfies these constraints, some degree of forward search is necessary. Otherwise, there is the obvious danger that obstacles would not be detected until they impeded vehicle advance. A more subtle danger is that a stance, at the limit of its support or kinematic constraints, may not be able to move into the next stance without violating a constraint.

Several abstractions are useful in planning within these constraints. The support polygon for a stance is the minimum bounding polygon, on the ground plane, that includes all leg-ground contact points. As long as the vehicle's center of force (approximately the center of gravity for a slow-moving vehicle) is held above the support polygon, the vehicle is statically stable. If five legs are on the ground while the sixth is recovering, the support polygon is generally a pentagon. If one of the five supporting legs fails, either due to mechanical failure or soil collapse, the support polygon would be reduced to a quadrilateral. Considering the failure of each of the legs in turn generates a number of support polygons equal to the number of supporting legs. The intersection of these polygons is the *conservative support polygon* (CSP) — the area that gives guaranteed static support even if any one of the supporting legs fails. When five legs are in contact with the ground, as during a leg recovery, the CSP is usually a pentagon. (Figure 4).

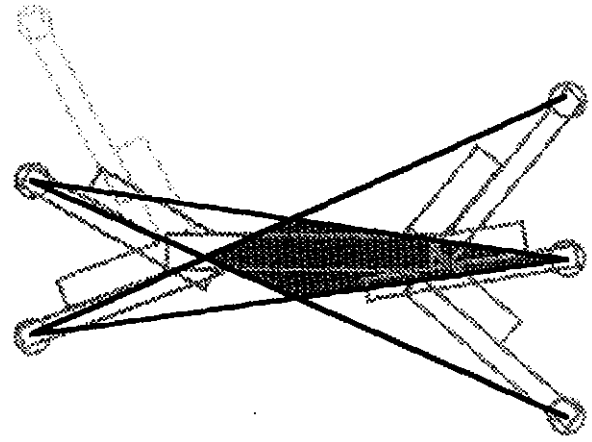


Figure 4 - Five Leg Conservative Support Polygon

When six legs are in contact with the ground, as during a body move, the CSP is generally a quadrilateral that subsumes all the five leg conservative support polygons. (Figure 5)

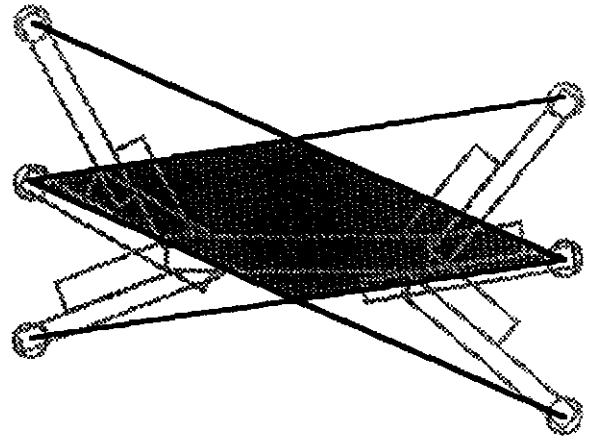


Figure 5 - Six Leg Conservative Support Polygon

The CSP abstraction is useful in the planning process because it provides limitations on the movement of the body and this in turn limits the footfalls that must be considered. (Figure 6) A feasible sequence of stable moves must chain together moves so that the body can glide from one CSP to the next.

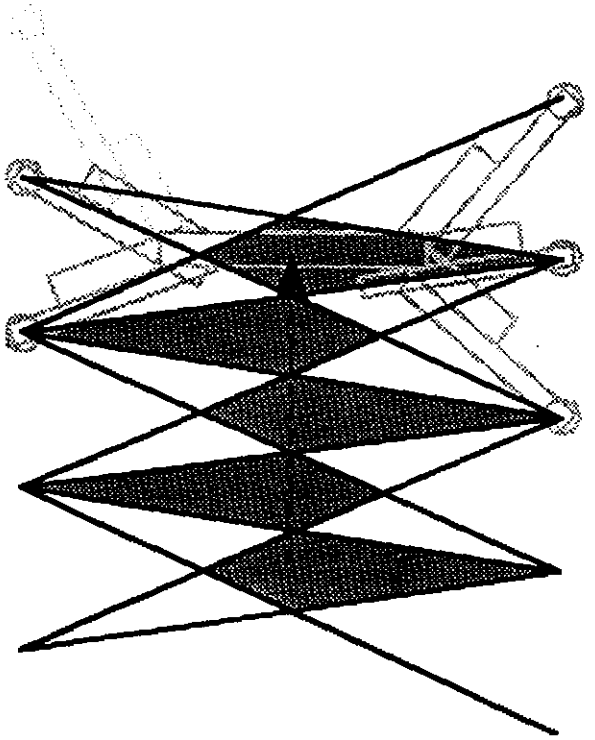


Figure 6 - Walking with continuous CSP security

To maintain productive advance the abstraction of reachable space for the leg stack simplifies the selection of the most productive rotation and translation. Each stack must be maintained in its reachable volume. (Figure 7) The crescent shaped regions are kinematically feasible based on the leg placements. Any translation and rotation that maintains each stack in its respective crescent is allowable. A search through this space quickly identifies the possible motion along any arc in space. The intersection of the CSP and stack reachability constraints limit the next body move.

Given the stack reachable space and CSP constraint abstractions there are two different and complementary ways of computing standard gaits: either as pattern of uniform stances which are modified based on the desired vehicle motion, or as a combination of evaluated features which cooperate to determine a preferred foot placement so that body motion may continue.

A uniform patterned circulating gait is set by fixing the pitch and tread at constant values. The difficulty is choosing values that optimize progress along a given trajectory. For example, by moving the legs in, close to the body laterally, and reaching far forward (narrow tread and long pitch) the body can propel the largest distance forward and backward. However, by stretching out to the sides with narrow spacing the between the legs (short pitch and wide tread) the greatest rotation is possible.

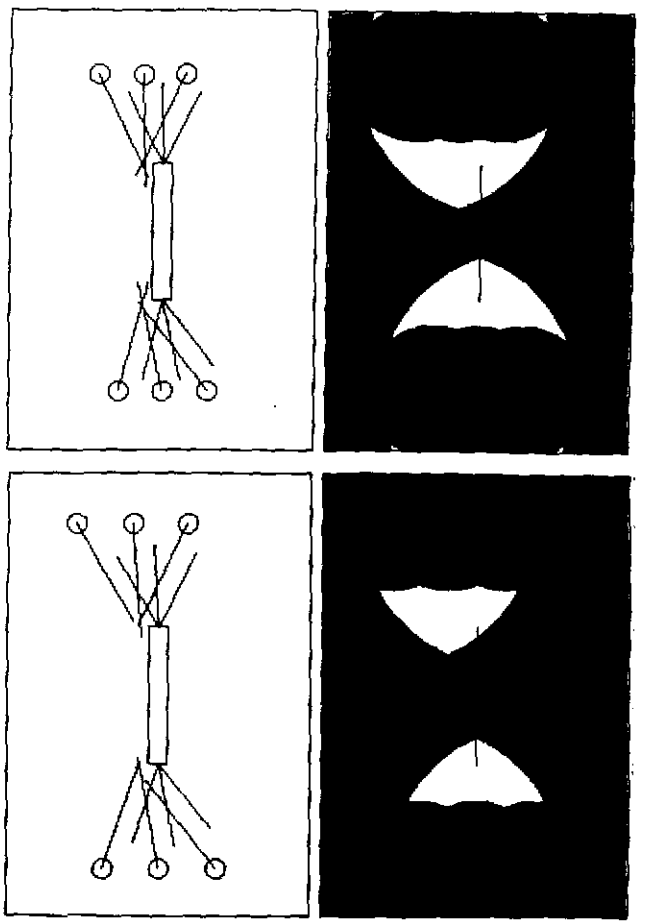


Figure 7 - Stack Reachable Space: Short Pitch/ Wide Tread and Long Pitch/Narrow Tread

Choosing constant pitch and tread lends itself to travel along an arc because as the body changes orientation along the arc the preferred footfalls distribute uniformly. Fixing the pitch and tread for walking along an arc means that the feet on the outside of the arc must be recovered more often than those on the inside. For arcs of small radius the forward pitch and tread of the stack to the inside of the turn are out of reach when the alternate backward pitch and tread are reachable so the action is to retrograde on the inside. Turns that are nearly centered on one set of feet would bind on each other and require occasional shuffle steps.

This strategy of adjusting the pitch and tread values based on the intended motion and using the stack-reachable space and CSP to constrain each move has demonstrated the generalized arc walking that was desired for the Ambler.

The advantage of the uniform gait approach is that planning is greatly simplified — the number of options to consider at each leg move is reduced — and progress along the trajectory is near optimal. The algorithm specifies the ideal locations of footfalls on moderate terrain. If there are obstacles, or if the legs start in an irregular configuration,

other constraints are considered as well. The ideal footfalls, plus the other constraints are algebraically combined as utility functions by considering each factor that impedes body advance.

There are four ways in which the legs can constrain body motion: (1) conflicts among the legs, (2) conflicts between a leg and the body, (3) leg kinematic limits, or (4) when the body exceeds its conservative support polygon. When one of these constraints is reached, the planner evaluates metrics to determine where to recover the appropriate leg. Metrics for the *local* advance (the advance over the next body move), the *global* advance (the advance possible until the recovered leg must be recovered again), and the area of the support polygon are normalized and multiplied to determine each potential leg move's relative merit. (Figure 8) These constraints, when combined, determine a footfall that maintains vehicle stability and advance. These utility metrics are all very local in nature; thus, the gait produced at this stage is locally optimal, but may be globally sub-optimal. Determining the proper foot to recover is simple in the cases of kinematic limits since the offending leg can be identified directly.

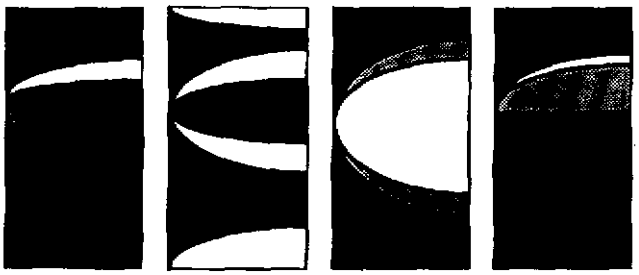


Figure 8 - Leg Placement (in configuration space): Global Advance, Local Advance, Area and Combined Cost Maps

When legs interfere, the leg with the larger unobstructed shoulder swing is chosen from the legs involved. Because the Ambler uses a circular gait, recovering this leg generally produces a larger body advance.

Through stages of refinement the Ambler's gait planner generates a uniform gait through easily travesable terrain but still selects acceptable leg and body moves in difficult terrain or transitions.

Footfalls

The gait selected by the gait planner prescribes the preferred leg placement area and provides reasonable assurance that a good footfall will exist. Within this area, approximately a square meter, it is necessary to examine the terrain and vehicle configuration to select the exact point of terrain contact. By combining features such as terrain roughness, required reach of the leg, and the complexity of the step in a geometrical representation the footfall planner selects the best footfall location.

There are a number of potential evaluation criteria for selecting acceptable footfalls. Considerations for a single footfall can be classified as pertaining to either the robot configuration or the terrain characteristics. For example, a walking robot just as a standard robotic manipulator, has workspaces that it can reach. Given the gait, a fixed reachable space is defined. This reachable space constrains the set of possible footfalls. Terrain features such as the slope and traction of the terrain can also be evaluated to provide constraint to the planning problem.

Constraints can be classified into discrete and continuous. (Table 1) Discrete constraints are those that *must* be met by a potential footfall. For example, a footfall must be within reach of the robot at the time the leg is recovered. Continuous constraints are those that have a scaled effect and therefore can be minimized to identify least constrained, or most desirable, footfalls.

	Robot Configuration	Terrain Features
Discrete	Maximum Reach Minimum Reach	Maximum Elevation Minimum Elevation
Continuous	Body Advance Visibility	Roughness

Table 1 - Implemented Footfall Constraints

The strategy for footfall planning is that discrete constraints threshold the evaluation space and then continuous constraints are combined in a adjustable weighting function. This weighting function combines measures of reachability, visibility, and potential body advance robot configuration constraints with elevation and roughness terrain feature constraints to produce a relative scaling, or cost map, of the appropriateness of each footfall. (Figure 9)

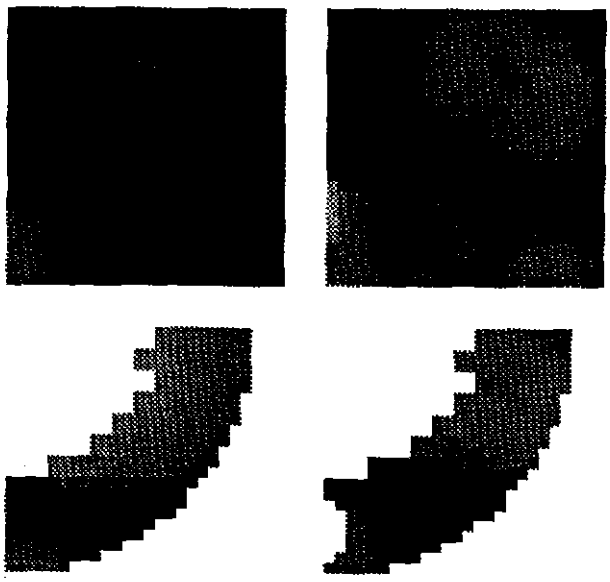


Figure 9 (clockwise from upper left) - Terrain elevation, terrain flatness, and body advance evaluations and the combined cost map.

The resultant cost map of the terrain locations scales the relative appropriateness of each footfall. The best location to place the foot can then be selected from the map.

Results

The implementation of planners has proceeded from the bottom of the hierarchy upwards. We have tested various forms of these planners in three environments, a single-leg test environment, a series of simulators and on the Ambler itself.

A test environment for a single Ambler leg was built with an overhead carriage hung from ceiling mounted rails. [Simmons89] The footfall planner has proven highly capable in commanding the leg to step through terrain strewn with rocks and meter-scale obstacles. The carriage can be pulled along once the leg is planted and a test terrain ten meters long has been traversed. Experiments in the test terrain guided the development of the footfall cost maps and proved that the footfall planner can advance through difficult rough terrain.

A simulator of the six-legged Ambler, implemented on a graphics supercomputer, has provided preliminary results of the effectiveness of the gait planning strategies. [Thomas90] The uniform gait approach has been successfully demonstrated as have cases of the utility based gait planning. Neither planner has been integrated with the footfall planner and tested in rough terrain. Work is ongoing to integrate the planning strategies into a single planner that generates smooth gaits with smooth transitions.

A preliminary gait planner has been used to generate flat floor walking with the Ambler. The implementation of planning software on the Ambler itself is ongoing. Qualitative results will be forthcoming.

Conclusion

We have identified a method of trajectory planning that, through a feasible traversability measure, identifies passable terrain. Further development of this planning strategy will allow identification the most desirable specific path.

The Ambler's gait planning approach develops uniform gaits for general arcs though moderate terrain regions and refines to acceptable leg and body moves in very complex terrain and maneuvering situations. With this strategy a robust and fast gait planning system has enabled the Ambler to walk in simulation and take its first steps in the real world.

Finally, we have demonstrated a footfall planner that can combine both robot configuration and terrain constraints to select terrain contacts. This planner has been proven in rough terrain and will fall into the hierarchy with the Ambler's gait planner as the vehicle begins to walk through rough terrain.

Acknowledgements

We would like to thank all the members of the Planetary Rover project at Carnegie Mellon. Most of them have passed along an idea or note on a scrap of paper at one time or another. In particular, Swami Mahalingham began the work in gait planning, Ben Martin and Joe Hirsch worked on early simulations, and Eric Krotkov and Reid Simmons are leading the effort to integrate the Ambler software system.

We gratefully acknowledge the support for this work provided by NASA under contract NAGW-1175.

Bibliography

[Bares89] Bares,J., et al., "Ambler: An Autonomous Rover for Planetary Exploration", in IEEE Computer, Vol. 22, No. 6. 1989.

[Bares90] Bares,J., Whittaker,W., "Walking Robot with a Circulating Gait," IEEE Int'l Workshop on Intelligent Robots an Systems (IROS)'90, July 1990.

[Brooks90] Presentation to the Carnegie Mellon School of Computer Science Distinguished Lecture Series, March 1990.

[Donner84] Donner,M., "Control of Walking: Local Control and Real Time Systems", Carnegie Mellon Department of Computer Science Technical Report CMU-CS-84-121, May 1984

[Kaneko88] Kaneko,M., Abe,M., Tanie,K., "Study on Walking Machines With Decoupled Freedoms," in the Journal of Robotics and Mechatronics, April 1988.

[Kwak88] Kwak,S., McGhee,R.B., "Rule-Based Motion Coordination for the Adaptive Suspension Vehicle," in Technical Report NPS52-88-011, Naval Postgraduate School, May 1988.

[Kweon90] Kweon,I.S., Kanade,T., "High Resolution Terrain Maps from Multiple Sensor Data," IEEE Int'l Workshop on Intelligent Robots an Systems (IROS)'90, July 1990.

[Lee88] Lee, W.J. and Orin, D.E. "The Kinematics of Motion Planning Over Uneven Terrain", IEEE Journal of Robotics and Automation, April 1988, pp. 204-212.

[Mahalingam89] Mahalingam, S., Whittaker,W., "Terrain Adaptive Gaits For Walkers with Completely Overlapping Work Spaces", in the Proceedings of Robots 13, Gaithersburg, MD, pp.1-14, May 1989.

[Rosenblatt88] Kiersey, D.M., Rosenblatt,J.K., Payton, D.W., "Autonomous Navigation in Cross-Country Terrain," Proceedings of the 1988 Image Understanding Workshop, pp. 411-419.

[Simmons89] Simmons,R., Mitchell,T., "A Task Control Architecture for Mobile Robots," in the

Proceedings of AAAI Spring Symposium, Stanford, CA, March 1989.

[Song88] Song,S., Waldron, K., "Machines that Walk: The Adaptive Suspension Vehicle," MIT Press, Cambridge, MA, 1988.

[Stentz89] Stentz, A., "Multiresolution Constraint Modeling for Mobile Robot Planning", Proceedings of SPIE Symposium on Advances in Intelligent Robotics Systems, November, 1989.

[Thomas90] Thomas,H., Wettergreen,D., Thorpe,C., "Simulation of the Ambler Environment", Modeling and Simulation Conference, Pittsburgh, PA, May 1990.

A perception and manipulation system for collecting rock samples[1]

T. Choi, H. Delingette, M. DeLuise, Y. Hsin, M. Hebert, K. Ikeuchi

The Robotics Institute,
Carnegie Mellon University,
Pittsburgh Pa15213

Abstract

An important goal of a planetary exploration mission is to collect and analyze surface samples. As part of the CMU Ambler project, we are investigating techniques for collecting samples using a robot arm and a range sensor. The aim of this work is to make the sample collection operation fully autonomous. We describe in this paper the components of the experimental system that we have developed, including a perception module that extracts objects of interest from range images and produces models of their shapes, and a manipulation module that enables the system to pick up the objects identified by the perception module. We have tested the system on a small testbed using natural terrain.

1 Introduction

One of the most important goals of a planetary exploration mission is to collect and analyze terrain samples. As part of the CMU Ambler project [2], we are investigating techniques for autonomously collecting samples. We have developed a system that is able to collect small rocks using computer vision and planning. Our goal is to eventually integrate the system to the Ambler system, a six-legged autonomous robot for planetary exploration.

We have developed a rock sampling system that includes: a robot arm, a range finder, and a small terrain mock-up that contains sand and small rocks. The goal of the rock sampling system is to identify, locate, and pickup rocks from the terrain. The control flow of the rock sampling system is shown in Figure 2: First an range image of the scene is taken and features are extracted from the image (Section 2). The features are surface features such as surface discontinuities that are used to extract the object boundaries. Then the contours of the objects in the scene are extracted. Since, we are dealing with natural environments, we make very weak assumptions on the possible shapes of the objects and on the distribution of the features in the image. To handle those constraints, we have developed a new shape extraction algorithm (Section 3.1) based on the concept of deformable contours. The set of points enclosed by the contour of an object is approximated by a superquadric surface (Section 3.2). In some cases the object representation using superquadrics may not be sufficient. An algorithm based on deformable surfaces can extract directly a surface representation of an object using the image features without relying on superquadric fitting (Section 4). Finally, the parameters of the surface that approximate each object (superquadric or deformable surface) are used to grasp it using a clam-shell gripper (Section 6). The algorithms for object extraction assume that there is an initial guess of the positions of the objects in the image. We present an algorithm for selecting the object location hypothesis automatically in Section 5.

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

2 Image acquisition and feature extraction

In order to manipulate objects, we need an accurate description of their shape. This implies that we need to use a sensor that can sense the 3-D surfaces observed in a scene. Therefore, the only possibility is to use a sensor that measures range data. Many range sensing techniques are available [3]. The range sensor that we are currently using is an active sensor that consists of a projector equipped with a computer-controlled LCD screen and a camera [14]. The projector illuminates the scene through the LCD screen. As several illuminations patterns are projected, the corresponding images of the scene are collected by the camera. The range to each point in the scene is recovered from the shape of the projected patterns. The output of the sensor is set of four 256×256 images: an intensity image and three images, X, Y, and Z that contain the three coordinates of the three spatial coordinates of each pixel. The coordinates are with respect to a fixed reference frame defined at calibration time. The spatial and range resolution of this sensor is appropriate for this application in which we need high-resolution measurements at close range. We currently use the intensity image for only display purposes although it could also be used in the object extraction algorithms [12].

Figures 3 and 4 show the images of two scenes. The upper left image is the intensity image, the other three images are the coordinate images. The coordinate images are coded on 16 bits and displayed on 8 bits which accounts for the periodic effect in those images. Figure 5 shows a 3-D display of the data from Figure 3.

Once an image is acquired, the next step is to extract features of the terrain that can help extract the objects of interest in the environment. Many different types of features can be extracted from range data [4] ranging from planar facets to local extrema of the principal curvatures [5]. However, most of those techniques do not apply to this problem mainly because we are working in an unconstrained natural environment which rules out all the feature types, *e.g.* planar or quadratic patches, that assume a known geometric structure of the environment. Furthermore, it is our experience that the standard techniques based on curvature analysis perform well only when the data is very accurate and well distributed. We have chosen an approach in which we detect local features that are relatively insensitive to noise. We do not force the features to provide a complete description of the terrain, in particular we do not expect those features to connect to each other to form the boundaries of the objects in the scene. Instead, we want each feature to give partial evidence of the presence of an object in its vicinity. Grouping the detected features into objects is the job of the segmentation algorithms introduced in the next Section.

Three types of features are extracted:

- Range shadows: Objects produce shadows in the range images, which are areas of the scene that are illuminated by the projector but that are not visible from the camera because they are occluded by an object's surface. This phenomenon occurs with any sensor that uses a triangulation technique. Range shadows are therefore a important cue for the extraction of objects. Extracting the range shadows does not require any image processing since they are identified by the sensor itself.
- Surface discontinuities: A surface discontinuity is a large variation of range between neighboring points in the image. Such discontinuities occur mostly in the vicinity of the occluding edge of an object. Surface discontinuities are detected by applying an edge detector to the image of the range values, $r = \sqrt{x^2 + y^2 + z^2}$. The final edges are obtained by thresholding the resulting edge magnitude. The threshold is computed from the distribution of the edge magnitudes in a large window centered at each image pixel. The reason for using a variable threshold is that the range r varies more rapidly as points are measured further from the sensor. Spurious edges would be detected if a fixed threshold were used.
- Surface normal discontinuities: Surface normal discontinuities occur when two surfaces intersect as is the case when an object is resting on top of the terrain. The normal discontinuities are detected by first computing the unit surface normal n at each point and by finding the low values of the dot products $n_1 \cdot n_2$ of the surface normals at adjacent pixels. The three coordinate images must be smoothed first since the surface normal computation is quite sensitive to noise in the data. Further smoothing is applied to the surface normals.

The image pixels that are labeled as one of the three feature types are grouped into connected regions. The set of feature regions is the input to the segmentation algorithms. Figure 6 shows the features computed from the image of Figure 3. The features are shown as shaded regions. As expected, the features are concentrated around the objects although some are detected on the underlying terrain and no group of features form a closed object boundary.

3 Object extraction: deformable contours and superquadrics

The features give an indication of where the boundaries of the objects may be located in the scene. However, the raw features are not sufficient for reliably extracting the objects from the scene because the objects may be small or partially buried in the terrain. Therefore, we cannot use a simple region extraction that would assume that the features are grouped into closed boundaries. Instead, we used the concept of deformable contours and deformable surfaces. The idea is that a contour that is attracted by 2-D forces generated by the detected features and by the data points measured on the terrain is iteratively deformed until the forces applied to it are in equilibrium. A smoothness constraint is added to the forces so that the contour or the surface does not have sharp discontinuities of orientation or curvature. The final product is a smooth contour that approximates the shape of an object that is partially enclosed by features. The advantage of this approach is that object descriptions can be extracted from the image even if only few scattered features are observed. This is in sharp contrast with other vision problems such as model-based object recognition in which an accurate model of the objects is known apriori. We do not make any assumption on the shape of the objects other than a maximum and minimum object size, and we do not make any assumption on the configuration of the features.

This approach is inspired from Witkin's "snakes" [11] and from Terzopoulos' symmetry-seeking surfaces [16]. We describe in detail the deformable contours algorithm in the next Section.The algorithm assumes that one point that lies inside the object is initially selected. The actual selection of this starting point is the object of Section 5. We assume for now that this point is available. Once a contour is extracted, a three-dimensional model of the corresponding set of points must be built. We use superquadrics to represent the object models (Section 3.2).

3.1 Deformable contours

A deformable contour is a contour in a range image that is subject to forces that change its shape over time. The contour reaches a stable shape when all the forces are in equilibrium. The points that are inside the region enclosed by the final contour are used to described the shape of the object. The algorithm used to derive a shape representation from the region is described in Section 3.2.

We represent a contour by an ordered set of pixel (r_i, c_i) where r_i is the row coordinate in the image, and c_i is the column coordinate. A 3-vector p_i, that is the position of the scene point measured at pixel (r_i, c_i), is associated with each pixel. In addition, the normal to the contour n_i is defined at each p_i. The n_i's are two-dimensional vectors expressed in image coordinates. Furthermore, n_i is always oriented from the inside to the outside of the contour. It is always possible to define such an orientation since the contour is guaranteed to be closed without self-intersections. Each p_i is subject to a set of forces. Each force is a signed scalar that indicates in which direction p_i is attracted. A positive force indicates that p_i is attracted toward the outside of the contour in the direction of the nearest feature. The algorithm is designed in such a way that the contour can only grow outward.

Each pixel of the contour is subject to two types of forces 7(a). The external forces are exerted by entities that are not part of the contour such as features. The internal forces depend on the contour itself and are independent of the data. Internal forces are typically used to force the contour to be as smooth as possible.

The first external force is generated by the features. It is an attractive force defined at each point p by:

$$F_{\text{feature}} = \sigma_{\text{feature}}\left(\frac{\|p - \mathcal{F}(p)\|}{R_{max}}\right) \qquad (1)$$

where $\mathcal{F}(p)$ is the point of the image features that is the closest to p, σ_{feature} is a function that relates the force to the distance between contour point and feature (Figure 7(b)), and R_{max} is the maximum expected object size. The closest point $\mathcal{F}(p)$ is calculated by searching the feature points along 16 directions around the contour normal. Since this is a potentially expensive operation, we use several constraints to limit the search: First, the features that are too far from the contour point are not considered. Second, we use the fact that the order in which features appear around an object is defined by the geometry of the sensor and can be computed beforehand thus eliminating features that cannot be part of the current object.

The second type of external force is generated by the starting point. Its purpose is to prevent the contour from "overgrowing" by generating an attractive force towards the center point. The force is defined by:

$$F_{\text{center}} = \sigma_{\text{center}}\left(\frac{\|p - p_0\|}{R_{max}}\right) \qquad (2)$$

where R_{max} is defined as before, p_0 is the starting point, and σ_{center} is the attraction function (Figure 7(c)).

The purpose of the internal force is to guarantee that the contour is reasonably smooth. The idea is to make the shape of the contour close to an ellipse. To do that we approximate the contour by an ellipse $\mathcal{E}$ of equation $(p - p_{\mathcal{E}})^t A(p - p_{\mathcal{E}}) = 1$, where $p_{\mathcal{E}}$ is the center of the ellipse, and A is a 2×2 symmetrical matrix. The distance between p and $\mathcal{E}$ is defined by:

$$D(p, \mathcal{E}) = \frac{|(p - p_{\mathcal{E}})^t A(p - p_{\mathcal{E}}) - 1|}{2\|A(p - p_{\mathcal{E}})\|} \tag{3}$$

$D(p, \mathcal{E})$ is an approximation of the Euclidian distance between p and $\mathcal{E}$. The internal force is defined by:

$$F_{\text{internal}} = \sigma_{\text{internal}}\left(\frac{D(p, \mathcal{E})}{K}\right) \tag{4}$$

where σ_{internal} is the attraction function (Figure 7(d)), and K is a constant that controls how far from an ellipse the contour is allowed to be. In practice $K = 0.4$.

The contour deforms itself iteratively. At each iteration, the internal and external forces are computed at each point. Each point is moved according to the resulting force, The complete algorithm follows two steps:

1. Initialize: The initial contour is a small contour centered at the starting point.
2. Iterate: The following steps are iterated until the contour does not deform itself significantly.
   - At each point p_i compute the sum of the forces: $F = F_{\text{feature}} + F_{\text{center}} + F_{\text{internal}}$.
   - p_i is moved by one pixel in the direction of the nearest feature point $\mathcal{F}(p_i)$ if $F > 0$.
   - Resample the contour after all the contour points have been moved according to the forces.
   - Estimate the best-fit ellipse $\mathcal{E}$.

Provided that there is a reasonable starting point, this algorithm produces object contours that are quite good approximations of the true object contour even if the features are very sparse. Figure 8 shows the regions that have been found for each object in the scene using the feature of Figure 6. The starting points were selected automatically using the algorithm of Section 5.

3.2 Superquadrics

Once regions corresponding to objects have been segmented out using the deformable contour algorithm, the corresponding set of 3-D points must be grouped into a surface representation. The resulting object models are used to compute grasp position and manipulator motion.

Although one could use the set of 3-D points computed by the segmentation directly, we use superquadrics to represent the objects. Superquadrics are generalizations of quadric surfaces [1] that can represent a wide variety of shape. Using superquadrics present several advantages: First, it is a compact representation that allows us to represent a wide range of surfaces using a small set of parameters. Second, it provides a global representation of an object whose surface is only partially visible. Lastly, the parameters of a superquadric surface are easily recovered from the coordinates of a set of points.

Superquadrics are described by an implicit equation $F(x, y, z) = 1$, where:

$$F(x, y, z) = \left(\left(\left(\frac{X}{a_1}\right)^{\frac{2}{\varepsilon_2}} + \left(\frac{Y}{a_2}\right)^{\frac{2}{\varepsilon_2}}\right)^{\frac{\varepsilon_2}{\varepsilon_1}} + \left(\frac{Z}{a_3}\right)^{\frac{2}{\varepsilon_1}}\right)^{\varepsilon_1} \tag{5}$$

and where (X, Y, Z) are the coordinates of (x, y, z) after transformation by a rigid transformation that defines the position and orientation of the superquadric, and a_1, a_2, and a_3 are the sizes of the superquadric along the three directions. Superquadrics can represent a variety of shapes form cubes to ellipsoids by varying the two "roundness" parameters ε_1 and ε_2. Other parameters such as bending and tapering can be included in the equation. To recover the superquadric from a set of points, we use the Levenberg-Marquart minimization approach suggested by Solina [1]. In this approach, the input set of points is first approximated by an ellipsoid which constitutes the starting point of the minimization, then an error function of the form:

$$E = \sum_{(x,y,z)\ \text{data point}} (F(x, y, z) - 1)^2 \tag{6}$$

is minimized with respect to the parameters of the superquadric. This approach works well in our case in which a dense set of points is measured on a portion of the surface (see [13] or [9]) for other superquadric fitting techniques).

Figure 9 shows the superquadric models of the objects found in Figure 8. The models are displayed as wireframes superimposed on the intensity image.

4 Object extraction: deformable surfaces

Deformable contours extract the objects by using essentially the geometry of the scene in the image plane. The result is a region in the image that has to be processed further to yield a complete description of the object. A more direct, although more costly, approach would be to directly find the closed surface that best approximates the data, that is the 3-D points measured on the terrain and the detected features. This leads to the idea of deformable surfaces which are smooth closed surfaces that are subject to forces from the terrain and the features. As with the deformable contours, the surface deforms itself until it closely fits the observed shape. The advantage is that the resulting closed surface should provide all the information needed to pick up the object. As in the case of deformable contours, the algorithm assumes that an initial point is selected inside each object.

The algorithm operates on discrete data, images and discrete features. However, for the sake of clarity it is best to think first of the case of a *continuous* deformable surface that is subject to forces and deforms itself over time. It can be shown that such a surface would reach a stable equilibrium when the Lagrangian of the system of forces reaches a minimum according to the principle of least action [8]. A similar application of the principle can be found in [17]. The Lagrangian is defined by: $L = T - U$ where T is the integral of the kinetic energy over time and U is the integral of the potential energy. If the surface is parametrized as $x = x(\eta, \omega, t)$, $y = y(\eta, \omega, t)$, $z = z(\eta, \omega, t)$, where t is the time, and (η, ω) are the parameters of the surface, then the problem is to find the function that minimizes L. This is a variational problem that can be solved by applying Euler's equation. To simplify the notations, we will denote the points of the surface by $r(\eta, \omega, t)$, r being the 3-vector (x, y, z), and we will denote the partial derivatives by using subscripts (*e.g.* $r_\omega = \frac{\partial r}{\partial \omega}$). Furthermore, we assume that the parameters η and ω vary between 0 and 1.

The term T depend only on the kinetic energy and can be written as:

$$T = \int_0^{t_0} \int_0^1 \int_0^1 \mu \|r_t\|^2 d\omega d\eta dt \tag{7}$$

where μ is a weighting factor that characterizes the inertia of the surface.

In our case, the surface should be deformed so that the following constraints are satisfied: The surface should be smooth, the surface should be as close as possible to the surrounding features, and the surface should be close to the points measured on the terrain. To satisfy those constraints, the potential energy term U is decomposed into three components:

$$U = U_{\text{smoothness}} + U_{\text{features}} + U_{\text{terrain}} \tag{8}$$

The term $U_{\text{smoothness}}$ encapsulates the constraint that the surface should be smooth and continuous. Formally it is defined by:

$$U_{\text{smoothness}} = \int_0^{t_0}\int_0^1\int_0^1 \omega_1\left(\|r_\omega\|^2 + \|r_\eta\|^2\right) + \omega_2\left(\|r_{\omega\omega}\|^2 + \|r_{\eta\eta}\|^2 + 2\|r_{\omega\eta}\|^2\right) d\omega d\eta dt \tag{9}$$

The weights ω_1 and ω_2 control how much importance is given to the smoothness constraint. The surface can have any arbitrary shape if they are equal to zero, on the other hand contributions from the features and the terrain are ignored if they are very large.

The term U_{features} implements the constraint that the surface should be as close as possible to the surrounding features. In order to define it, we first have to define the distance between a point and a feature. In order to do that, we represent each feature $\mathcal{F}$ by the 3-D polygonal approximation of its skeleton which is a set of 3-D line segments. The distance between a point r on the deformable surface and a feature $\mathcal{F}$ is the distance between r and its projection on the set on line segments that describes $\mathcal{F}$. We denote the projection by $r(\mathcal{F})$. Strictly speaking, we should compute the distance between r and all the points of $\mathcal{F}$. Since this is too demanding computationally, we use the polygonal approximation which allows us to compute the projection directly. With this definition of $r(\mathcal{F})$, we define:

$$U_{\text{features}} = K\int_0^{t_0}\int_0^1\int_0^1 \sum_{\mathcal{F}} S(r,t,\mathcal{F})\|r - r(\mathcal{F})\|^2 d\omega d\eta dt \tag{10}$$

where the sum is taken over all the features $\mathcal{F}$, and where K is a weighting factor. If we think of a set of springs linking each point of the surface to each feature, K would be the stiffness of the springs. The attraction exerted by the features is basically proportional to the squared distance between the point and the feature $\|r - r(\mathcal{F})\|^2$. An attenuation factor $S(r,t,\mathcal{F})$ is added to avoid one undesirable effect of the pure spring model: points that are very far from the features are always subject to very strong forces. What we would like instead is to have a strong attraction to all the points to initiate the deformation and to have the strength of the attraction decrease over time. For a given distance $\|r - r(\mathcal{F})\|$, this is equivalent to vary the stiffness of the spring as a function of time. Furthermore, it is undesirable for the features to apply an arbitrary large force to the points that are far away. We need a cutoff distance over which points are not attracted. We define the correction factor by:

$$S(r,t,\mathcal{F}) = \sigma\left(1 - \frac{t_0}{t_0 - t}\frac{\|r - r(\mathcal{F})\|^2}{r_0^2}\right) \tag{11}$$

where the function σ varies from 0 at $-\infty$ to 1 at $+\infty$ (Figure 10). σ implements the idea of a cutoff distance: points that are too far away from the feature are not attracted. The cutoff distance is given by the normalizing term r_0. In addition to the cutoff distance, for a given distance $\|r - r(\mathcal{F})\|$, the term $\frac{t_0}{t_0 - t}$ makes the stiffness of the spring vary over time: The spring is strong a $t = 0$ and weakens over time until it eventually disappears at $t = t_0$ at which time only the smoothness constraint and the attraction from the terrain are taken into account.

Another way to look at equation is to consider the term $K\|r - r(\mathcal{F})\|^2$ as a fitting term in that it forces the surface to be as close as possible to the features, and to consider the term $S(r,t,\mathcal{F})$ as a segmentation term in that it takes into account only the group of features that is close to the starting point. The last term of the potential U_{terrain} reflects the attraction between the surface and the terrain. It is defined as:

$$U_{\text{terrain}} = \beta\int_0^{t_0}\int_0^1\int_0^1 \frac{1}{\|r - r(T)\|} d\omega d\eta dt \tag{12}$$

where $r(T)$ is the data point that is closest to the surface point r. Since the term inside the integral would become arbitrarily large as the surface moves closer to the data, we introduce a cutoff distance D at which the potential stops increasing. The potential is therefore redefined as:

$$\begin{cases} \frac{1}{\|r - r(T)\|} & \text{if } \|r - r(T)\| \geq D \\ \frac{1}{D} & \text{if } \|r - r(T)\| \leq D \end{cases} \tag{13}$$

This potential implements a gravity force that increases as points move closer. This has the effect that the feature term is dominant initially when the surface is far from the observed terrain while the terrain becomes dominant as the surface moves closer to the terrain. Notice that strictly speaking we should take into account the contributions from *all* the data points in the computation of the force applied to a single surface point. Since this is computationally untractable we limit ourselves to the closest data point.

We now have a definition of the function L given a set of features and a set of points measured on the terrain. The problem is now to find the surface $r(\eta,\omega,t)$ that minimizes L. The solution is found by straightforward application of Euler's equation. We obtain the differential equation:

$$\mu r_{tt} = \frac{1}{2}\beta P(r) + \omega_1(r_{\omega\omega} + r_{\eta\eta}) - \omega_2(r_{\omega\omega\omega\omega} + 2r_{\omega\omega\eta\eta} + r_{\eta\eta\eta\eta}) + KS(r,t,\mathcal{F})(r - r(\mathcal{F})) + KS_1(r,t,\mathcal{F})\|r - r(\mathcal{F})\|^2 \tag{14}$$

where S_1 is computed from the first derivative of σ: $S_1(r,t,\mathcal{F}) = -\sigma'\left(1 - \frac{t_0}{t_0 - t}\frac{\|r - r(\mathcal{F})\|^2}{r_0^2}\right)\frac{t_0}{t_0 - t}\frac{r - r(\mathcal{F})}{r_0^2}$, and P is the gradient of the potential due to the terrain attraction, that is the integrand of U_{terrain}: $P(r) = \frac{r - r(T)}{\|r - r(T)\|^3}$.

Applying Euler's equation solves the problem in the case of a *continuous* surface subject to the attraction of the features and the terrain and to a smoothness constraint. To actually compute a solution to the resulting differential equation, we need to construct a discrete approximation of both the surface, that is a discretization of the parameter space (η,ω) and of the time t. Let us consider first the case of the parameter space. Using a straightforward discretization of η and ω in regular intervals of $[0,1]$ would lead to serious problems at the edges of the parameter space just like sampling a sphere along the meridians and parallels leads to problems at the two poles. Since it is not desirable to be forced to handle special cases in the discretization, we would like to use a representation of the parameter space that is as uniform as possible. To do that, we first create a unit sphere that is tesselated using the icosahedron decomposition [6, 7], each point M_i of the tesselation is parametrized by its spherical coordinates (η_i,ω_i) and is a sample point on the surface. The tesselation of the sphere has the property that it is very uniform and that it does not exhibit any poles. With this representation the integrals become sums over the sample points, for example the integral with respect to (η,ω) in U_{terrain} becomes:

$$\sum_{M_i \text{ sample point}} \frac{1}{\|r(\eta_i,\omega_i) - r(T)\|} \tag{15}$$

The time axis is also discretized: the deformation of the surface is implemented as an iterative process, the discrete time is simply the iteration number. With those discrete representations of (η, ω) and t, the derivatives involved in the final solution are approximated by the appropriate finite differences. In particular r_{tt} is given by a combination of the values of r at iterations $t, t-1$, and $t+1$: $r_{tt} = r(t+1) - 2r(t) + r(t-1)$. Replacing r_{tt} by its discrete approximation in the differential equation, we can express the surface at iteration $t+1$, that is the vectors $r(\eta_i, \omega_i)$, as a function of the surface at the two previous iterations t and $t-1$. If F is the right-hand side of 14, we have:

$$r(t+1) = r(t) + (r(t) - r(t-1)) + F \tag{16}$$

After initialization, the deformable surface is iteratively updated using this relation.

To summarize, the algorithm can divided into two steps:

1. Initialize:
   - Extract the terrain features: shadows, discontinuities, normal discontinuities. Compute the polygonal approximations of the skeleton of the features.
   - Generate the discretization of the parameter space by computing a uniform sampling (η_i, ω_i) of the unit sphere.
   - Generate an initial surface. The initial surface is a sphere, that is $r_i = C + Ru_i$, where C is the starting point that is inside the object, R is the radius of the smallest object that we expect to extract, and $u_i = (r_i - C)/\|r_i - C\|$. The algorithm for selecting C is described in Section 5.
2. Iterate until the number of iteration is greater than t_0
   - For each point r of the surface, compute the projections $r(\mathcal{F})$ and $r(\mathcal{T})$.
   - Compute the derivatives of r with respect to η and ω using finite differences.
   - Compute the update term F using Equation 14.
   - Update the surface using Equation 16.

The result of the deformable surface algorithm is illustrated in Figure 11: The upper left part of the figure shows the features overlaid in white on top of the intensity image of a small scene. The upper right part shows a 3-D view of the terrain with the polygonal approximations of the features. The bottom three images show the evolution of the shape of the approximating surface as the algorithm proceeds.

5 Automatic object selection

We have assumed so far that a point is chosen inside each object to initiate the object segmentation process both in 2-D and 3-D. This point should be qualitatively "close to" the center of the object. The question of finding those initial points still remains. The simplest solution is to have an operator interactively select a point in the observed image. This would be acceptable in a teleoperated mode with the appropriate user interface. However, it would be more useful to be able to automatically compute the starting points from the input images. Since there is no prior constraint on where the objects may be in the scene, the only information that we can use are the features and a geometric model of the sensor. Specifically, the automatic segmentation is based on the observation that the presence of an object generates a shadow region in the range image. Therefore, the objects in the scene should be "near" the shadow regions extracted from the range image. The meaning of "near", that is the position of an object with respect to its shadow, is given by the sensor model.

The geometry of the problem is shown in Figure 12. For the sake of clarity, this geometry assumes a one-dimensional sensor; the reasoning can be extended without difficulties to a 2-D sensor: A projector P illuminates the scene while a camera C observes the illuminated scene. We assume that a sensor model provides the coordinates of P and C in a common coordinate system. An object in the scene creates a shadow region between points A and B, corresponding to illumination directions L_A and L_B that are known from the measured coordinates of A and B and from the sensor model. Based on this geometry, the occluding object must be within the dashed region R. A starting point for the 2- or 3-D snakes can be computed by taking the center of that region. It is important to note that this algorithm does not give us the center of the (unknown) object but rather a point that is enough inside the object for the object extraction algorithm to work.

The geometry is similar with a 2-D sensor except that the two points A and B are now contours. In practice, two corresponding points A and B are chosen on the shadow contour and the region R is identified using the 1-D geometry. The starting point S is selected within R at some nominal distance D from A. D is chosen based on the average expected radius of the objects in the scene and based on the minimum and maximum sizes of objects that we can handle given a gripper configuration. Those are reasonable criteria since there is no point in segmenting out objects that we cannot manipulate. D is also used to remove small shadow regions, presumably due to noise, and large regions, generated by objects too large to handle.

The key to automatic object extraction is an accurate geometric model of the sensor that allows us to compute the hypothesized position of objects in the scene based on observed shadow regions. We have implemented this technique using a model of our current sensor. However, it is important to note that the algorithm can be generalized to any range sensor provided that a geometric model exists. We are in the process of modifying the algorithm in order to use an existing geometric sensor modeling system [10]. This will lead to a largely sensor-independent segmentation program.

6 Manipulation

Once we have extracted object descriptions, either superquadrics or deformable surfaces, the last step is to grasp the object. Many different types of gripper design and grasping strategies are possible. The choice of a particular type of grasping is dictated by the analysis of the task. Assuming that the objects to be sampled are mostly isolated and are resting on a soft surface, *e.g.* sand, the grasping task has the following characteristics:

- The objects are far enough from each other. No collision occurs between the gripper and the neighboring rocks.
- We can allow the collision between the gripper and the neighboring sand. This is because
  - damaging the neighboring sand grains is not important,
  - the collision between the gripper and neighboring sand does not cause the configuration of the rock to change.
- we do not know the exact shape of a rock beforehand.

Based on the characteristics of the task and the possible grasping strategies [15], we have selected the spherical grasping strategy using a clam-shell gripper. The gripper has two hemispherical jaws that can close around the object. Using a surface representation of the objects, the grasping strategy is as follows: the center of the gripper is first aligned with the center of mass of the surface, then the gripper is rotated so that the jaws are parallel to the main axis of the surface. Finally the gripper is lowered until the jaws are in contact with the terrain surrounding the object. The object is grasped by closing the two jaws. Figure 13 show the gripper and the grasp operation.

This approach works well under the stated conditions. However, we need tighter control of the grasping operation than is provided by

the spherical grasping in more difficult environments (*e.g.* Figure 4). In this case, we will use the object model calculated from the deformable surfaces algorithm conjunction with a three-finger gripper. The object model is more accurate than the superquadric model, and the three-finger gripper allows for more flexibility in the grasping. The price to pay is in longer computation time, and in more complex gripper design and control.

7 Conclusion

We have developed a testbed for sampling in unstructured terrain, that is the identification and manipulation of small natural objects. We have implemented the complete cycle of perception, representation, and manipulation. The objects are extracted from range images from surface features using either deformable contours or deformable surfaces. The objects can be represented by superquadric surfaces and by discrete surfaces. The system has been demonstrated in real natural environment using a manipulator equipped with a clam-shell gripper.

Our current work concentrates on building a more complete description of the terrain by using multiple images, hierarchical representation of the observed scenes, and by using more accurate object description such as deformable surfaces. We are working on a three-finger gripper to perform manipulation in a cluttered environment. Finally, we are exploring strategies for modifying the terrain using the manipulator to facilitate the sampling operations.

The sampling system currently resides on a small testbed. We want to eventually move it to a real vehicle, and to demonstrate the interaction between navigation and sampling, thus providing a complete system for planetary exploration.

References

[1] R. Bajcsy and F. Solina. Three Dimensional-object Representation Revisited. Technical Report MS-CIS-87-19, Univeristy of Pennsylvania, Department of Computer and Information Science, March 1987.

[2] J. Bares, M. Hebert, T. Kanade, E. Krotkov, T. Mitchell, R. Simmons, and W. Whittaker. An Autonomous Rover for Exploring Mars. *IEEE Computer*, (6), June 1989.

[3] P.J. Besl. Range Imaging Sensors. Technical Report GMR-6090, Computer Science Department, GM Research Laboratories, Warren, Michigan, March 1988.

[4] P.J. Besl and R.C. Jain. Three-dimensional object recognition. *ACM Computing Surveys*, 17(1):75–145, March 1985.

[5] M. Brady, J. Ponce, A. Yuille, and H. Asada. Describing surfaces. In *Proc. 2nd International Symposium on Robotics Research*. MIT Press, Cambridge, MA, 1985.

[6] C. Brown. Fast display of well-tessellated surfaces. *Computer and Graphics*, 4(4):77–85, April 1979.

[7] J.D. Clinton. Advanced structural geometry studies, part I: polyhedral subdivision concepts for structural applications. Technical report, NASA, September 1971.

[8] C. Fox. *An introduction to the calculus of variations*. Dover Publications Inc., 1963.

[9] A. Gupta, L. Bogoni, and R. Bajcsy. Quantitative and Qualitative Measures for the Evaluation of the Superquadric Models. In *Proc. IEEE Workshop on Interpretation of 3-D Scenes*, November 1989.

[10] K. Ikeuchi and J. C. Robert. Modeling Sensor Detectability with VANTAGE Geometric/Sensor Modeler. In *Proc. of DARPA Image Understanding Workship*, pages 721–746. Science Application, Inc., May 1989. (a slightly longer version is available as CMU-CS-89-120).

[11] M. Kass, A. Witkin, and D. Terzopoulos. Snakes: Active Contour Models. *Intern. Journal of Computer Vision*, 2(1):321–331, 1988.

[12] Y.P. Lim. *Shape Recognition in the Rocks World*. PhD thesis, Dept. of Electrical Engineering and Computer Science, MIT, April 1988.

[13] A. Pentland. On the extraction of shape information from shading. Technical Report Vision science technical report 102, MIT Media Lab, March 1988.

[14] K. Sato, H. Yamamoto, and S. Inokuchi. Range imaging system utilizing nematic liquid crystal mask. In *International Conf. on Computer Vision*, pages 657–661, London, 1987.

[15] C.L. Taylor and R.J. Schwarz. The Anatomy and Mechanics of the Human Hand. In *Artificial Limbs*, pages 22–35, 1955.

[16] D. Terzopoulos, A. Witkin, and M. Kass. Symmetry-Seeking Models and 3D Object Recognition. *Intern. Journal of Computer Vision*, 1(1):211–221, 1987.

[17] S.W. Zucker, C. David, A. Dobbins, and L. Iverson. The organization of curve detection: coarse tangent fields and fine spline coverings. In *Proc. Second Intl. Conf, on Computer Vision*, Tampa, December 1988.

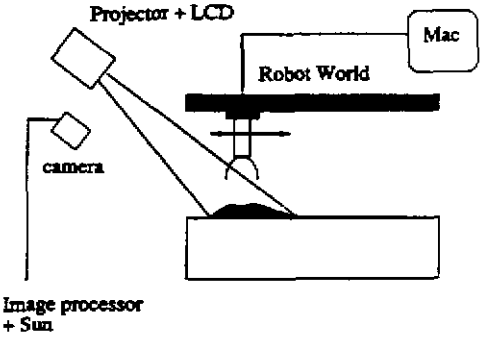


Figure 1: Testbed

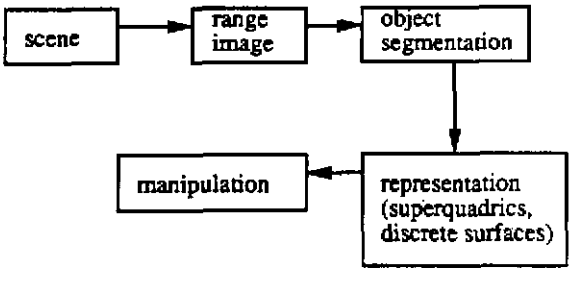


Figure 2: Control flow

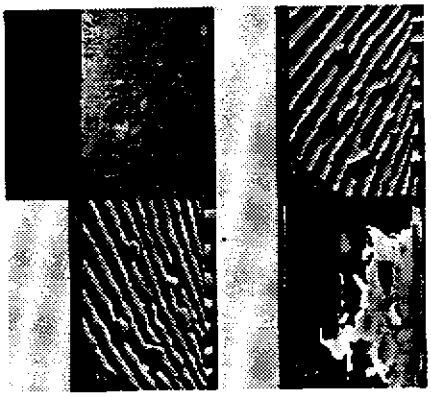


Figure 3: Range image of a scene

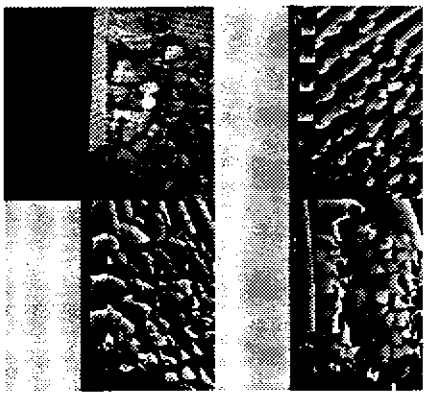
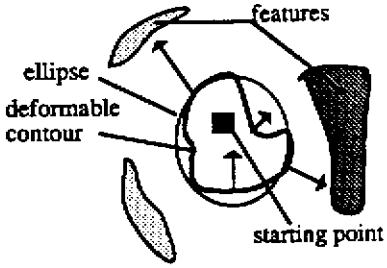


Figure 4: Range image of a complex scene

Figure 5: 3-D view of the data of Figure 3

Figure 6: Features extracted from the image of Fig. 3



(a) Forces applied to a deformable contour

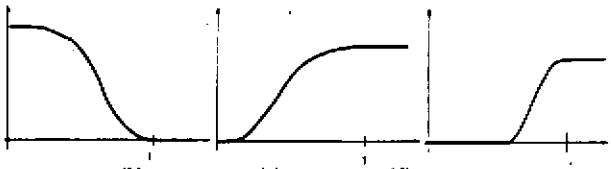


Figure 7: 2-D segmentation algorithm

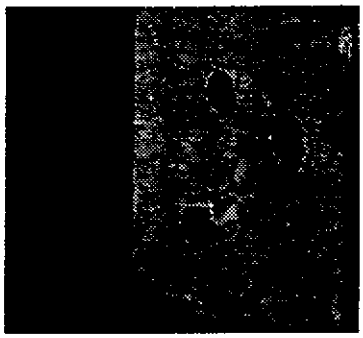


Figure 8: 2-D segmentation algorithm

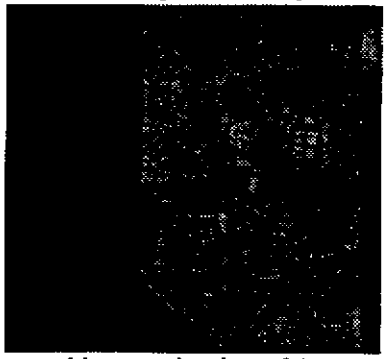


Figure 9: Superquadric approximations of the objects of Fig. 8

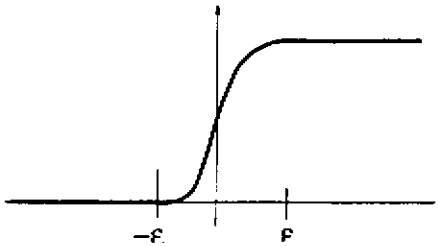
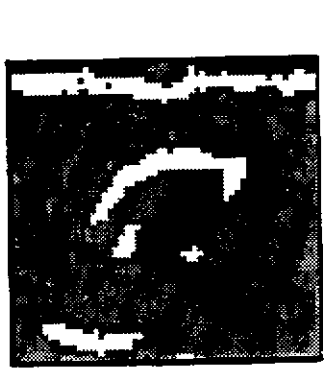
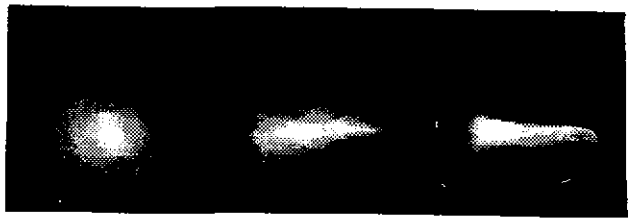


Figure 10: Sigma function



(a) Image of a scene and features

(b) 3-D view of the terrain



(c) Initial shape (d) Intermediate shape (e) Final result

Figure 11: 3-D segmentation algorithm

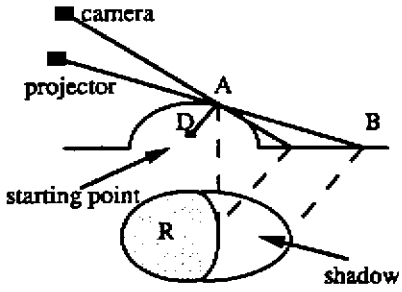
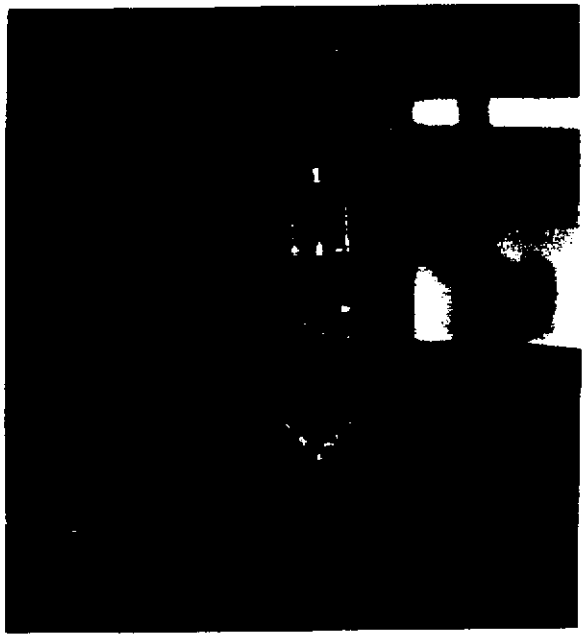


Figure 12: Automatic object selection



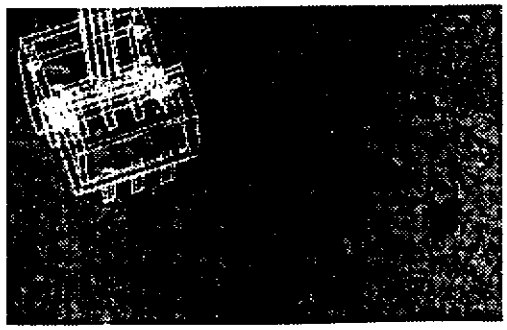


Figure 13: Gripper and grasp strategy