SAND REPORT
SAND2002-2596
Unlimited Release
Printed August 2002

Robust Planning for Autonomous Navigation of Mobile Robots In Unstructured, Dynamic Environments: An LDRD Final Report

G. Richard Eisler

Prepared by
Sandia National Laboratories
Albuquerque, New Mexico 87185 and Livermore, California 94550

Sandia is a multiprogram laboratory operated by Sandia Corporation, a Lockheed Martin Company, for the United States Department of Energy under Contract DE-AC04-94AL85000.

Sandia National Laboratories

Issued by Sandia National Laboratories, operated for the United States Department of Energy by Sandia Corporation.

Available to DOE and DOE contractors from
U.S. Department of Energy
Office of Scientific and Technical Information
P.O. Box 62
Oak Ridge, TN 37831

Telephone: (865)576-8401
Facsimile: (865)576-5728
E-Mail: reports@adonis.osti.gov
Online ordering: http://www.doe.gov/bridge

Available to the public from
U.S. Department of Commerce
National Technical Information Service
5285 Port Royal Rd
Springfield, VA 22161

Telephone: (800)553-6847
Facsimile: (703)605-6900
E-Mail: orders@ntis.fedworld.gov
Online order: http://www.ntis.gov/ordering.htm

SAND2002-2596
Unlimited Release
Printed August 2002

Robust Planning for Autonomous Navigation of Mobile Robots In Unstructured, Dynamic Environments: An LDRD Final Report

G. Richard Eisler
Satellite Data Processing
Sandia National Laboratories
P.O. Box 5800
Albuquerque, NM 87185-0973

Abstract

This report summarizes the analytical and experimental efforts for the Laboratory Directed Research and Development (LDRD) project entitled "Robust Planning for Autonomous Navigation of Mobile Robots In Unstructured, Dynamic Environments (AutoNav)". The project goal was to develop an algorithmic-driven, multi-spectral approach to point-to-point navigation characterized by: segmented on-board trajectory planning, self-contained operation without human support for mission duration, and the development of appropriate sensors and algorithms to navigate unattended. The project was partially successful in achieving gains in sensing, path planning, navigation, and guidance. One of three experimental platforms, the Minimalist Autonomous Testbed, used a repetitive sense-and-re-plan combination to demonstrate the majority of elements necessary for autonomous navigation. However, a critical goal for overall success in arbitrary terrain, that of developing a sensor that is able to distinguish true obstacles that need to be avoided as a function of vehicle scale, still needs substantial research to bring to fruition.

Acknowledgements

This author wishes to thank the following individuals for their invaluable contributions, time, extensive discussions on the project, and input to this final report:

Paul Klarer, Organization 15252, who developed the architecture for and supervised the development of the HAGAR vehicle,

John Harrington, 15252, for development and operation of the Minimalist Autonomous Testbed vehicle,

Jeff Bradley, Doug Jordan and Ted Kim, 2338, for developing the navigation analysis tools and the IMU package for land navigation

Wendy Amai and Emily Mitchell, 15252, for extensive software coding and troubleshooting of vehicle and basestation code

Daniel C De Baca, 15252, for hardware support of the HAGAR vehicle, and

Chris Lewis, 15211, for basestation and vehicle code development and support and Swarm RATLER support.

The author also wishes to thank Ken Jensen, 15212, Keith Miller, 15252, and Rush Robinett, 6200, for their support and extensive discussions on this project.

Table of contents

Table of figures

Introduction

Battlefield missions for mobile robots include providing military reconnaissance, surveillance, and target acquisition (RSTA). The search for capable configurations to handle the multitude of scenarios and obstacles for autonomous navigation to fulfill RSTA missions has spawned a rich field of possible solutions. Sandia's applications of collective swarms of mobile robots emphasize the decentralization of control and redundancy of capability, in order to accomplish tasks that a centralized command could not do. However the advantages of using many devices are tempered with the ability to find the appropriate mix of on-board intelligence and sensors so that the cost of manufacture or battlefield loss is not prohibitive.

For RSTA missions, an individual vehicle must take long and short range navigational viewpoints. The long viewpoint questions "Where am I in the world and what's the best course to traverse considerable distances as economically (from a power standpoint) and stealthily as possible?" Accurate maps are critical to the long range. The short viewpoint seeks to find a way around an inconveniently placed rock or tree, a meandering stream, or correct a planned course over a ledge that was not noted on a digital map.

The ultimate goal of this project is to provide an algorithmic-driven, multi-spectral approach to point-to-point navigation of a single, small mobile robot in unstructured environments. The end deliverable is to provide a rugged, cost effective and fieldable hardware/software system characterized by:

1. Segmented on-board planning, coupling internal algorithms with appropriate on and off-board sensors to solve long and short-range navigation problems,
2. Being self contained, that is operate with no human support for the mission duration, and
3. Being robust enough from an algorithm and sensor standpoint to "think" its way through truly unstructured environments.

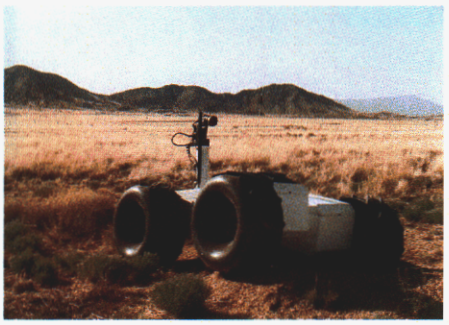


Figure 1. HAGAR Mobile Robot

To do this successfully, redundancies would be incorporated into both sensors and algorithms. The ability to "regroup" from inadequacies, loss, or failure of one method or device and "recapture" capability with another is crucial for "hands-off" operation.

During the pursuit of the aforementioned goals in this project, several major changes in research direction and technology shortcomings were noted. For instance, traditional on-board navigation was accomplished through a combination of GPS and dead-reckoning techniques, without metrics to gauge position accuracy. Control Subsytems Department 2338 was enlisted to add proven missile navigation techniques to the mix of traditional approaches as a means to provide quantitative measures of position accuracy and improvement. Computer code to manage vehicle hardware operations on the HAGAR

(High Agility Ground Assessment Robot [1]) was accomplished using a single-threaded DOS approach. This proved to be very "brittle" to changes and extensive discussions were held to chart new directions for standardizing hardware/software architectures to alleviate this problem [2]. Lastly the goal of autonomous navigation is critically dependent on being able to sense the environment for obstacles that would impede the progress of a vehicle as a function of scale. The emphasis on scale is important because obstacles are vastly different for a small vehicle as versus say, a battle tank. This goal demanded sensing technology for general off-road terrain beyond the current state of the art and limited the project's overall progress.

The Problem

The targeted unstructured terrain for the project is given in the Figure 2. This terrain is characterized by a lack of improved roads, rapidly changing elevation, diverse vegetation and surface texture, and obscured views. Unless blessed with an abundance of mobility, sensing, and/or knowledge, a typical mobile robot would have to employ optimal amounts of these to accomplish a successful traversal.

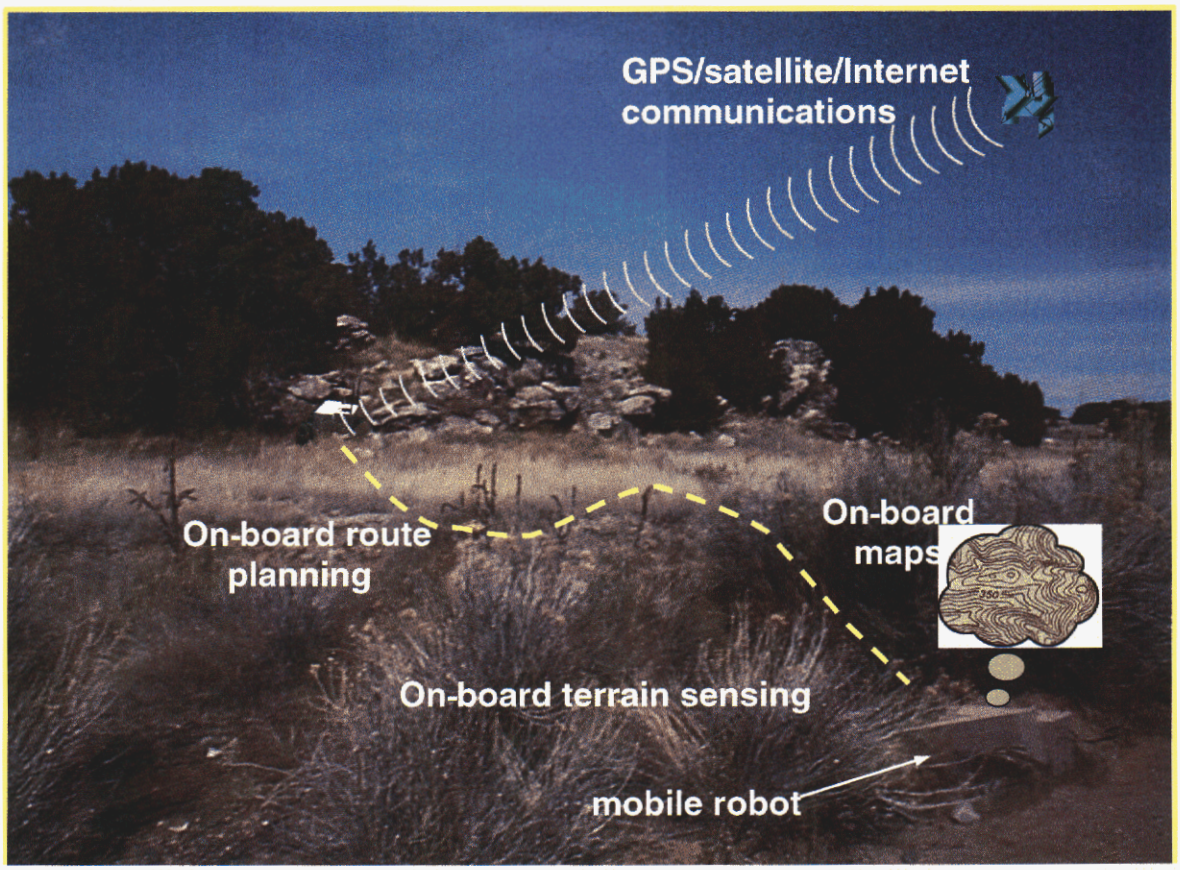


Figure 2. Autonomous Navigation for Unstructured Terrain

As displayed in the Figure 3, our interpretation is that mobility, sensing, and knowledge form the triad of capability that determines the level of autonomous navigation that is possible.

The project was organized to meet milestones of increasing difficulty: They are listed as follows with their associated risks.

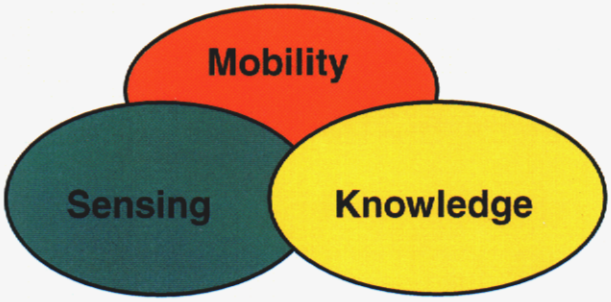


Figure 3. Overlapping capabilities needed for autonomous navigation

Year 1: Navigate autonomously over a pre-planned route

Risk - Planned vehicle route with GPS navigation may not be sufficiently accurate to maintain vehicle on known road.

Year 2: Navigate autonomously over an established road path that the vehicle plans on-board.

Risks: There may be unforeseen problems with sensing. Can't correlate sensed scenery with maps or update maps. On-board planned trajectories over arbitrary road terrain may present greater than anticipated vehicle power consumption

Year 3: Navigate autonomously off-road over a pre-planned route. Development of inertial navigation hardware and algorithms

Risk: Arbitrary terrain may prove non-negotiable for the sensor suite developed

Mobility

Three hardware platforms were used in this project to demonstrate various aspects and levels of autonomous navigation, a Swarm RATLER™ [1], HAGAR[2], and the MAT vehicle[4].

Swarm RATLER

The Swarm RATLER serves as the design archetype for a family of Sandia's teleoperational and autonomously operating electric vehicles (Figure 4). Eight RATLER™ vehicles have been built at Sandia as a test platform for cooperative control and sensing applications.
The RATLER design originated from a lunar exploration mission [3]. These electric, all-wheel-drive vehicles employ two composite bodies joined by a passive central pivot. This flexible structure when combined with an aggressive asymmetric tread on custom carbon composite wheels provides agile off road capabilities.

Figure 4. Swarm RATLER negotiating rocky outcropping

The RATLER vehicles are equipped with a PC104 form factor Intel 80486 processor for control. This computer interfaces to a wide range of sensors and peripherals. Software on the vehicles is currently a single-threaded DOS-based application for simplicity (which unfortunately would come to limit computational complexity and flexibility as discussed under

System Architecture). The vehicles have been programmed to operate either through teleoperation or autonomously. The RATLER vehicles rely heavily on Radio Frequency (RF) signals for communications. Currently, the vehicles are outfitted with differential GPS sensors, and two spread-spectrum RF modems. One modem is for inter-vehicle and base-to-vehicle communication and the other is for the differential GPS signal. Video cameras communicate to the base-station via a separate RF video link.

A laptop computer is used as the base-station. A Windows NT application was written by Chris Lewis, Dept. 15211 to control the vehicles from the base-station. A Graphical User Interface (GUI) displays vehicle status information and allows the operator to monitor the vehicles positions on a Geographic Information System (GIS) map – either aerial photo or topological data, as well as view live video from a selected vehicle (Figure 5).

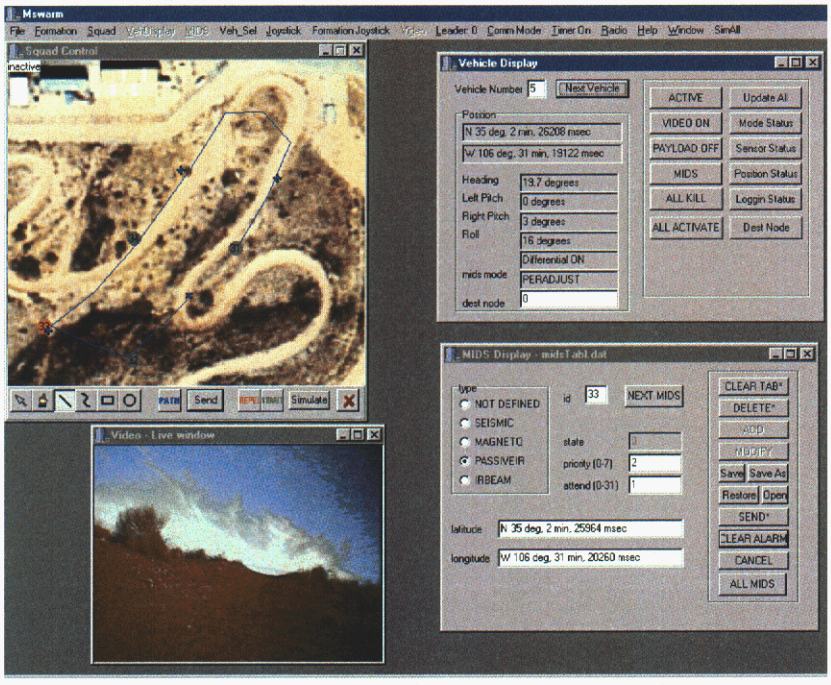


Figure 5. Swarm RATLER basestation display

Mission specific control modes such as teleoperation, formation following, autonomous navigation, and perimeter detection can be initiated and monitored using this GUI interface [10].

HAGAR

The High Agility Ground Assessment Robot (HAGAR) chassis, originally built in 1996 for use in unexploded ordnance recovery at the Sierra Army Depot in Reno Nevada, is a second-generation RATLER™ one. This mechanical chassis incorporates an improved all-carbon central pivot joint and an improved motor mount system [11]. HAGAR was refurbished for this project, and in the process was upgraded in the areas of sensing, computing, telemetry and control systems as used on the autonomous SWARM vehicle systems (Figure 6). HAGAR represents the third generation of full scale RATLER™ technology.

Each body half is equipped with a custom Faraday cage to house all electronics, primarily to eliminate emitted RF noise from the computing system and ensure as quiet an RF environment as practically possible. Inside one cage is a 233 MHz Pentium based CPU chosen for its upgrade potential to a real-time operating system (RTOS). In addition, the

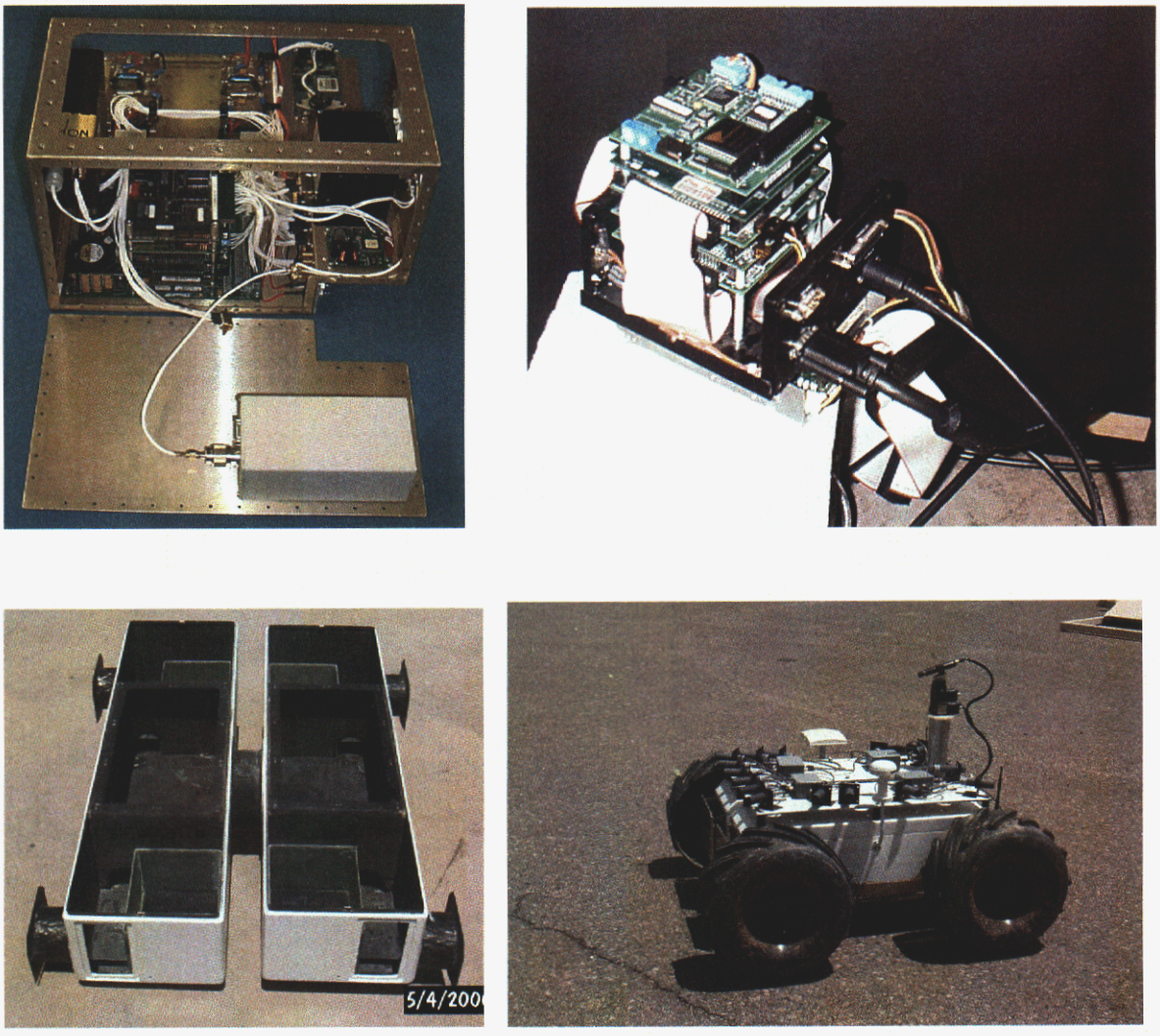


Figure 6. HAGAR components – clockwise from lower left – split body halves, Faraday cage with components, PC104 computer stack, and assembled 1-meter HAGAR vehicle

CPU has PC104 PLUS compatibility and is equipped with PCMCIA, 2 Mb of onboard removable FLASH, 100 Mbit Ethernet, 16 channels of 12 bit A/D, 16 channels of digital I/O, 2 channels of 12 bit D/A, and 10 RS232 serial ports. The sensor suite includes an electronic compass, pitch and roll inclinometers, high-resolution optical encoder odometers, differential GPS, and a 6-axis inertial rate sensor. Additional features of this vehicle include a high-resolution color CCD video camera on a powered pan/tilt unit, telemetry system capable of up to 115 kilo-baud, dual power, 2-watt video transmitter, and fully independent four-wheel electric drive. Finally, the second Faraday cage is

dedicated for use in supporting additional sensing and computing hardware to facilitate autonomous navigation capability development.

The onboard control software was ported from the Swarm RATLER environment to provide teleoperation and autonomous navigation capabilities with sensor based reflexive obstacle detection and avoidance capabilities [11].

MAT (Minimalist Autonomous Testbed)

A unique goal of the Minimalist Autonomous Testbed is to minimize computation and the use of complex sensors so that the complete system is scalable to a wide range of robotic platforms (small to large) at a low per unit cost [12]. MAT is a skid-steered, wheeled vehicle with a wheelbase and track of 1 foot (Figure 7). Wheel diameter is 9 inches and it is powered with two 12W, brushed DC motors. It weighs 25 pounds and is powered from a 12-volt, 7-ampere-hour lead acid battery that typically runs the entire vehicle for about four hours. Maximum speed is 5 inches per second.

Figure 7. MAT vehicle with bumper switches and WAAS GPS system

It is equipped with a WAAS-corrected GPS receiver, FreeWave 900 MHz two-way radio (9600 baud), a magnetic compass, pitch and roll inclinometer, odometer, and is controlled with an 80186 computer. The on-board computer, which is used only for interface purposes, is linked through the radio with an 80486 (75MHz) computer that handles all high-level processing. The MAT vehicle was developed by John Harrington of Dept. 15252, and expanded upon technologies developed for the SIR sentry vehicle [3,5] of the mid-1980's. From a cost standpoint, the MAT vehicle also demonstrated that considerable research capability could be assembled at a very attractive unit price (~$2K).

Obstacle Sensing

Obstacle sensing on board the three mobility platforms employed roll (about the longitudinal vehicle axis) and pitch (about lateral axis) tilt sensing which were used to trigger a reflexive avoidance maneuver, if a given vehicle body angular threshold was exceeded. HAGAR and MAT also employed bumper switches configured with square contact plates (Figure 8). On HAGAR, Hall-effect switches were the sensing mechanism and were slaved to provide a similar avoidance maneuver if compressed past a given threshold.

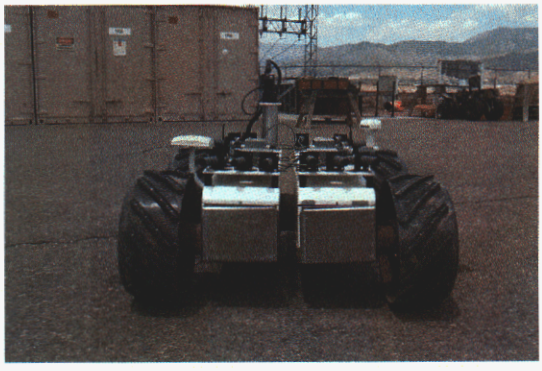


Figure 8. HAGAR vehicle showing bumper switch plates

The bumper switches on the MAT vehicle, located on the front and back of the vehicle, were the only obstacle sensor type. The force required to actuate these switches were set at a level that allowed the sensor to ignore grass but trigger on bushes and other dense objects that would impede vehicle movements.

The obstacle detection system for HAGAR consisted of two different types of sensors, the first being a bank of ultrasound sensors that can detect targets up to ten feet in distance. The second detection system consisted of a camera and a visible laser. One processor is used to process the data from both types.

Sixteen Massa M-5000 Smart ultrasonic sensors were placed in optimal positions around the robot (Figure 9). The M-5000 sensors generate a high frequency ultrasonic pulse, measure the time it takes for the echo to return, and calculate the target distance.

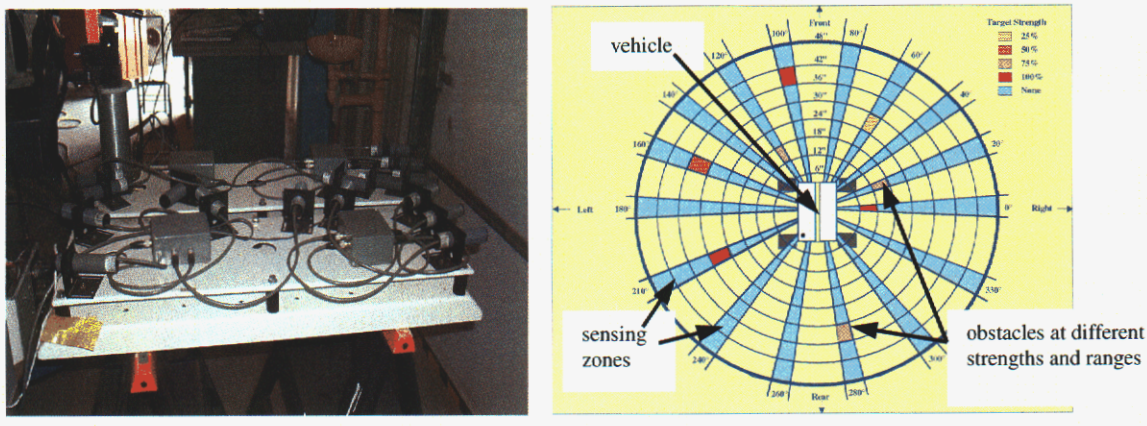


Figure 9. HAGAR mounting of Massa ultrasonic sensors with operational display

This distance information is output to a RS-484 communication port, which can be read into the PC104 stack. All of the sensors are daisy-chained on one serial port, and triggered by a single I/O line on the PC104 stack. To detect obstacles around the robot, all of the sensors are triggered at the same instant and data from each is read through the serial port. The data from each sensor is comprised of two items, 1) its distance to an obstacle which has been detected within a beam angle of 8° and 2) the associated signal strength. The distance is given in inches and the strength is quantized to 4 values, 25%, 50%, 75%, or 100%. The data received from the sensors is displayed on a polar grid describing what obstacles are detected within a 10ft. circular radius of the robot (Figure 9).

Numerous tests with the ultrasonic sensors showed that using the aforementioned output, the sensors were not able to discriminate "hard" objects, (i.e., large rocks, substantial bushes) that needed to be avoided, from "soft" ones (i.e., grasses) that could have been overrun. As such, obstacle avoidance with them was limited to well-defined objects in an uncluttered environment (i.e., traffic cones on pavement). The lack of a suitable "discriminating" sensor for arbitrary terrain content *severely limited the scope of the overall project* and the achievable milestones.

A vision detection system was developed and mounted on top of HAGAR with the Lasiris 50 milli-watt eye-safe laser about one foot above the Watec digital 8 bit camera (Figure 10). The Lasiris laser has a line projector with a thirty-degree fan angle placed in front of it . When pointed at the ground at an approximate 63-degree angle the projected line would cover forty-two inches across the ground at about 6 feet in front of the vehicle. A twelve-mm lens was focused on the ground approximately six feet out in front of the vehicle such that the entire laser line could be seen in the field of view. Positive and negative obstacles could be identified and characterized by the deflection of the line as it appears in different pixels in the image collected by the camera. Keeping this device correctly calibrated on smooth dirt roads proved difficult, and as such, was used only to a very limited extent.

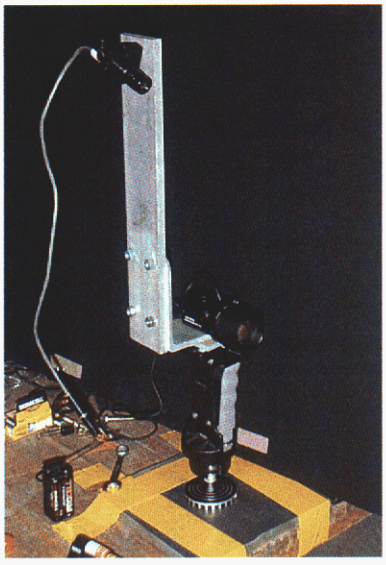


Figure 10. Lasiris laser configuration

From an operational standpoint for the ranging sensors (such as the ultrasonics), once sensed, obstacles had to be tracked unless they could be identified on a single "hit". The tracking demanded that several hits were recorded on a given stationary obstacle to insure that identification of said obstacle was: 1) the true one and not some artifact of vehicle attitude, such as reflections from the ground which would disappear as vehicle attitude changed, and 2) to insure that a stationary obstacle was resolved to the same location

given that it may be tracked by different sensors on the moving vehicle. A rectangular grid was implemented for tracking on HAGAR. Figure 11 shows a simulation of a point-to-point traversal in a two-dimensional field of obstacles and a tracking-grid extraction of obstacles within a 15-unit detection radius of the vehicle. The simulation employs a potential field guidance scheme [see Guidance and Control] and no path planning other than a linear attractor force to the finish point. The ratio of detection radius/vehicle radius was 10. In Figure 11, the plot on the left shows the complete simulated field of variable size obstacles. The one on the right is the simulated detection and grid assignment of those parts of the obstacles that would be identified within the sensing zones as depicted in the polar plot shown in Figure 9. Note that the potential field guidance does not produce the most direct route to the finish.

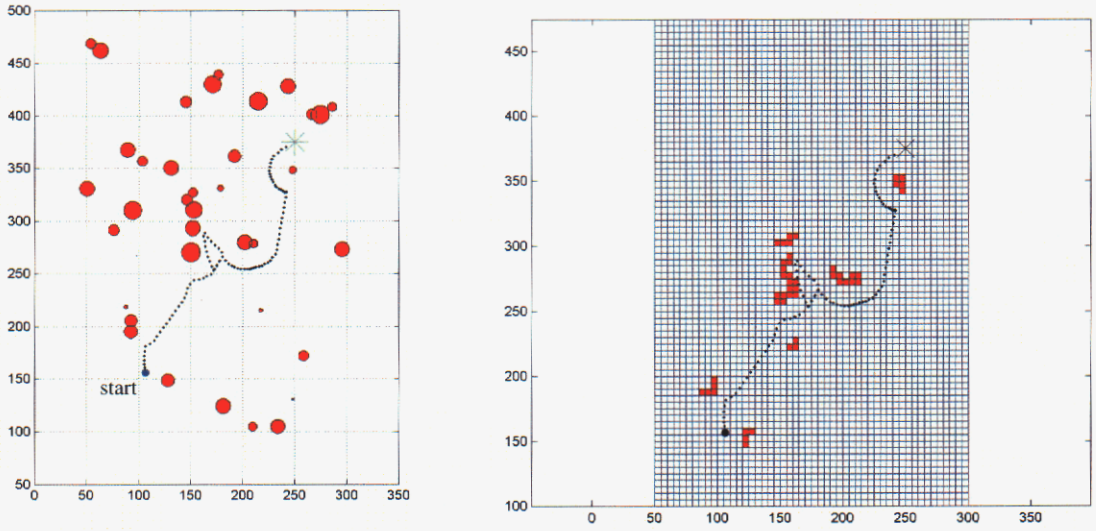


Figure 11. Simulation of autonomous traversal with obstacle sensing and tracking extraction

The MAT vehicle was able to construct an obstacle map without the need to track by using the bumper switch contact to assign the "contacted" obstacle location to the laptop, off-board map at the current vehicle position (computed from a combination of GPS and dead-reckoning).

Knowledge

Knowledge developments addressed the areas of path planning and digital map usage, guidance algorithms, and navigation.

Route Planning

A route planner uses a priori information about the environment that is to be traversed to plan a sequential set of locations to be visited. Subordinate to this is the need to identify major obstacles to avoid and/or natural corridors that should be followed. While the latter component is not necessary for success, its use can maximize the vehicle's chances of completing a path while minimizing the time to traverse a path that otherwise might involve considerable exploration. Generating paths from data obtained off-board (i.e., a digital map) implies that the path coordinates can be related to true ground positions and accurately followed by the vehicle [12].

Obstacle detection/classification/avoidance is responsible for defining a traversable path. The first step of this function is to detect and localize the object followed by classification. Classification results in determining whether the object should be avoided or ignored. Avoidance involves planning a path around the object that ultimately leads to the final destination [12].

A route planning system was developed by Chris Jones of Dept. 15222 to allow the user to analyze a segment of terrain from a Digital Elevation Model (DEM) data file of terrain at Sandia's Robotic Vehicle Range (RVR) as shown on the right of Figure 12. The path-planning environment allows the user to interactively visualize the 3D terrain model and plan minimum cost paths across the terrain based on metrics of mission requirements or goals.

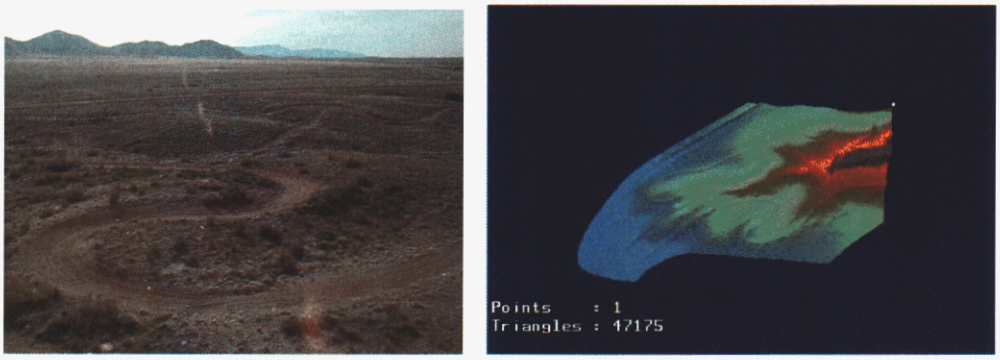


Figure 12. Robotic Vehicle Range motocross terrain and corresponding DEM reconstruction

(A MATLAB [13] capability to extract DEM maps from SAR data for analysis can be obtained from William Hensley of Dept. 2348).

The underlying planning algorithm uses a weighted graph approach, either a uniform grid or a quadtree representation. Each graph node represents a unique position of the robot in the environment and each graph edge represents a path segment connecting the edge's two end points. Each edge is assigned a weight representing the cost for the robot to traverse the edge. The edge weight is determined by a number of factors (slider set on the left of Figure 13), including: length, roll, pitch, radio communications, covertness, and path width (used for error margins). Each weight factor is assigned an importance factor, set by the user in accordance with mission requirements or goals, which determines how much it will influence the overall edge weight. Once the graph is built, any number of paths through the environment may be quickly extracted. A standard weighted graph search, such as Dijkstra's algorithm or A*, returns the minimum cost path from the start to goal position [8] (Figure 14). In cases where dynamic replanning may be performed due to unexpected obstacles during path traversal, an algorithm such as D* is more suitable [9]. This path-planning environment was able to give the user the option of selecting from the above listed search algorithms (Dijkstra, A*, D*).

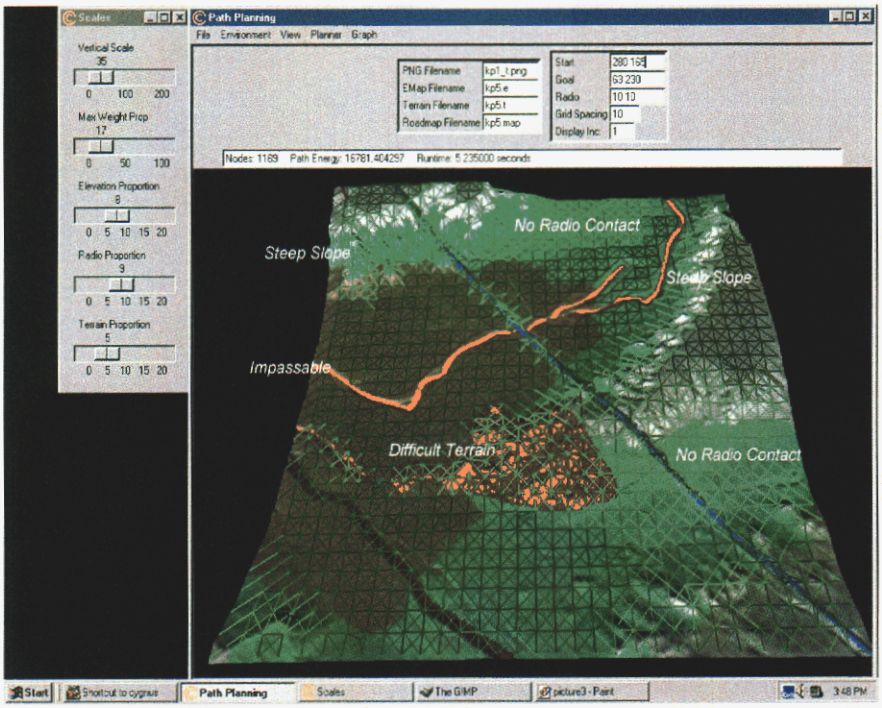


Figure 13. Path planner showing grid vertices, sensitive area, and roads for a DEM

Terrain models were used to determine the graph edge weight components (e.g. terrain pitch, line-of-sight radio communication). Supported terrain models include: USGS DEMs, SAR, and a variety of custom data formats. High-resolution aerial photographs (black and white, color infrared, etc) could also be overlaid on top of the terrain model to aid in distinguishing vegetation type and density as well as cultural features such as roads, trails, and buildings.

This path planning environment allowed the user to interactively view the 3D terrain models and any planned path(s). Regions of visibility may be highlighted, (note red and

green areas in Figure 13), to aid in analysis of the terrain or any paths. (This is useful for seeing "visibility" regions of a line-of-sight radio tower or areas visible by an enemy location if covertness is desired). Selecting start and goal positions for path planning, as well as radio tower positions, enemy locations, and obstacle locations were included.

Comparison of search algorithms: Dijkstra's algorithm and A* are very similar. The only difference is that A* uses a predictive measure of estimating cost to the goal that can help focus the direction of the search. This feature of A*, however, can also make it less efficient in some environments, including many natural terrain environments, as estimating cost to the goal is not always practical. Both Dijkstra's algorithm and A* (as well as D*) will all return a minimum cost path. In some cases, they will return different physical paths because it happens that there are multiple paths through the environment of the same minimal cost value. The D* algorithm allows for dynamic replanning in the presence of unexpected obstacles during path traversal. D* is essentially a modified version of A*. One other note is that the search algorithms also returned and notified the user if no path was available between the selected start and goal points.

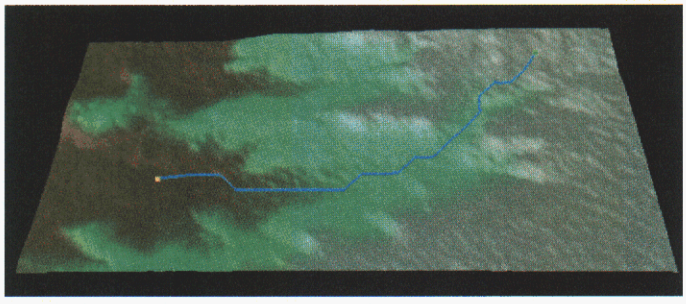


Figure 14. Planned path for 1-kilometer descending traverse minimizing distance and terrain slope using Dijkstra's algorithm on DTED level 4 for Sandia's Remote Vehicle Range Terrain

A pseudo-code description of the basic Dijkstra graph search to develop shortest paths on ordered grids follows:

```
DijkstraAlg(weighted graph, vertex start)
  for all verticies V
    currDist(V) = INFINITY

  currDist(start) = 0
  toBeChecked = all verticies
  while toBeChecked not empty
    V = a vertex in toBeChecked with minimal currDist(V)
    remove V from toBeChecked
    for all verticies U adjacent to V and in toBeChecked
      if currDist(U) > currDist(V) + weight(VU)
        currDist(U) = currDist(V) + weight(VU)
        predecessor(U) = V
```

The MAT vehicle environment uses a Dijkstra path-planning algorithm for both a priori objects and objects detected by on-board sensors [12]. The algorithm is a breadth-first search with no additional heuristics. Currently, this is not a limitation because only small areas are being searched. This algorithm is quite efficient, being able to search an array of 150^2 elements in approximately one second.

MAT operation begins by selecting a previously stored base map that indicates known objects. If no map exists, a map that is void of objects is used. The vehicle is then teleoperated to a starting point in the area of operation. A map icon is then positioned to reflect this physical location. At this point, the map reflects the actual position of the MAT vehicle. An autonomous operation is initiated by moving another map icon to a desired destination within the mapped area. The route planner finds the shortest path from MAT's current position to the destination while avoiding all known objects. The complete path may involve intermediate waypoints if objects are encountered. The computer begins following the route by decomposing the path into actual motor commands and sending them to the vehicle via radio. If additional objects are encountered along the way, MAT stops and enters the new object's location in its memory map. The object is noted as a one-foot square bounded by a one-foot boundary around the object. This creates clearance between the object and any path that is subsequently planned. Using the updated map, the entire path is re-planned from the current position to the destination. Excessive slope, as detected by the inclinometers, is also processed as an object. In this case, the object is drawn as a line five feet long and parallel to the slope's contour. This process continues until MAT reaches its final destination [12]. `Figure 15` demonstrates MAT's ability to solve a shortest path maze problem (i.e., obstacles known beforehand).

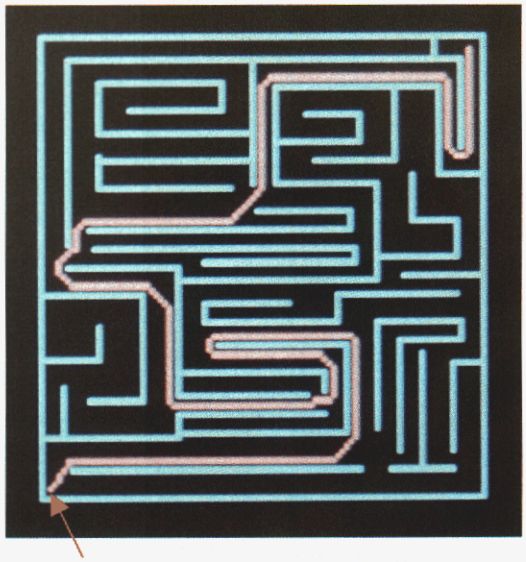
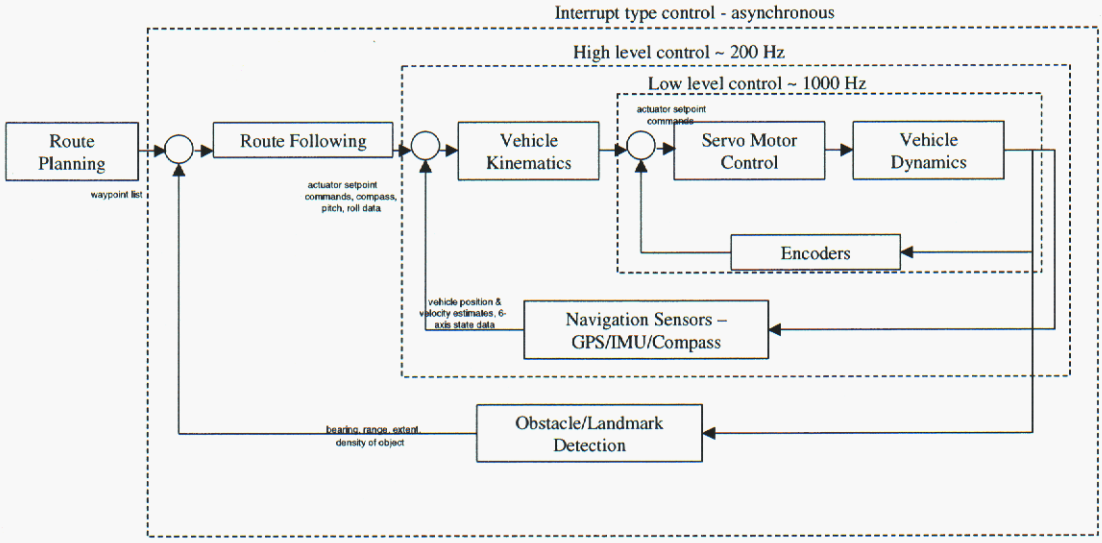


Figure 15. MAT's path planning solution to finding the shortest distance through a maze

Guidance and Control

From a controls' standpoint, the RATLER vehicle architecture is depicted in Figure 16. On the RATLER family of vehicles, the Route Planning "reference input" is a stored table of waypoints. The guidance scheme (contained in the High Level Control loop) to follow the waypoint list is encased in a "state machine" and provides a feedback algorithm driven by position and heading errors between waypoints. Obstacle detection at the current state of development can "change the vehicle state" from this high level guidance to initiate a reflexive avoidance maneuver followed by a return to the waypoint navigation state once the obstacle has been cleared.



Ref: David. Wilson, Dept. 15211, December 2001

Figure 16. RATLER controls block diagram

Guidance (or Route Following/Vehicle Kinematics) comprises that task that generates wheel speeds that enable accurate following of the planned path while avoiding obstacles for which there was no advance planning. The guidance law for inter-waypoint travel solves the system [6]

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{bmatrix} = \frac{1}{2} \begin{bmatrix} r\cos(\theta) & r\cos(\theta) \\ r\sin(\theta) & r\sin(\theta) \\ \frac{r}{l} & -\frac{r}{l} \end{bmatrix} \begin{bmatrix} \omega_r \\ \omega_l \end{bmatrix} = B \begin{bmatrix} \omega_r \\ \omega_l \end{bmatrix} \tag{1}$$

where x, y are the relative downrange and crossrange states, l is the distance from the center of the vehicle to the wheel, r is the wheel radius, and θ is the compass direction. The differences between desired (subscript d below) and actual location and orientation are used as feedback as follows:

$$\begin{bmatrix} e_x \\ e_y \\ e_\theta \end{bmatrix} = \begin{bmatrix} x_d - x \\ y_d - y \\ k_1\left(\theta - a\tan 2\left(y_d - y, x_d - x\right)\right) + k_2\left(\theta - \theta_d\right) \end{bmatrix} \tag{2}$$

where e_x, e_y, e_θ are the state errors which "drive" the guidance algorithm, and k_1 and k_2 are gains. The wheel velocity commands are determined by

$$\begin{bmatrix} \omega_r \\ \omega_l \end{bmatrix} = B^+ \begin{bmatrix} e_x \\ e_y \\ e_\theta \end{bmatrix}. \tag{3}$$

Potential field guidance methods were also studied for effectiveness. These employ virtual attractor and repellor forces from sensed obstacles and target goals on the vehicle that vary as $1/d^n$, where d is the vehicle-to-attractor/repellor distance and n is an arbitrary tuning constant. A simulation was developed in MATLAB to demonstrate the efficacy of this method in a dense, randomly distributed and sized field of obstacles. The following plots show successful and faulty transit cases (Figure 17).

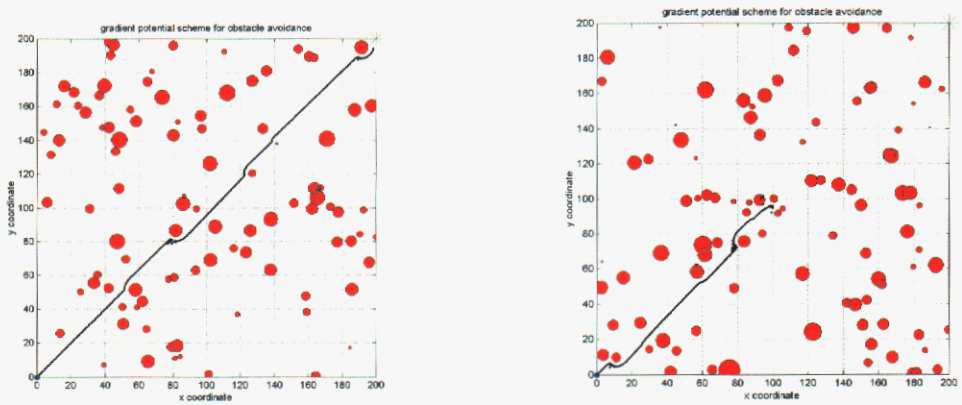


Figure 17. Comparison of successful and faulty potential field guidance attempts in a simulated 2-D obstacle field

There is no advance path planning and the simulated vehicle simply picks its way between diagonal corners of the map based on a linear attractor (i.e., proportional to d) at the upper right hand corner point and inverse square repellors from all obstacles within a given detector range (20 units for this case). The plot on the left demonstrates a successful transit due to the lack of encountering any concave-shaped arrays of obstacles. The vehicle is successful in being repelled from the obstacles along its path while being draw to the upper right corner target. On the right, a concave grouping of obstacles is encountered which causes attractor and repellor forces to align, providing a stable equilibrium from which there is no escape.

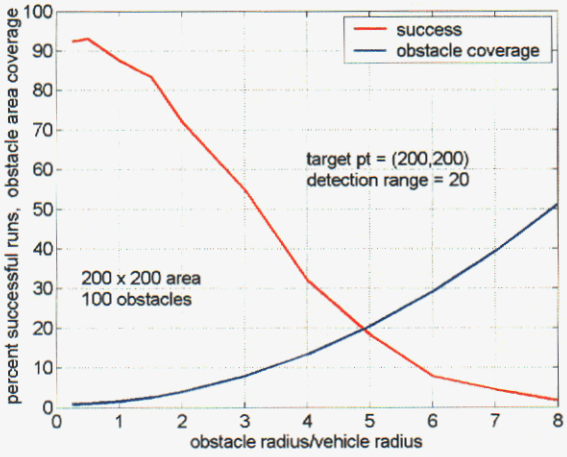


Figure 18. Statistical study of potential guidance effectiveness as a function of increasing obstacle/vehicle size

A statistical study based on this guidance method was compiled. For a given set of mean obstacle/vehicle size ratios, 500 runs were performed at each ratio value. The two-dimensional terrain and detection range were the same as in Figure 17. The results are shown in Figure 18. The percent of successful completion exceeds 90% for small relative obstacle sizes and diminishes to 20% for obstacle area coverage of 20% of the terrain.

This plot demonstrates that it would behoove the user to operate within a realm where the obstacles are no larger than the vehicle to maintain a credible (>90%) success rate with this guidance method. Obviously, this becomes less of a constraint as the vehicle size grows.

Navigation

Both analytical and experimental efforts were provided by Control Subsystems Department 2338 to quantify navigation errors from a variety of sources. Dept 2338 would develop an aided inertial navigation package consisting of on-hand, low cost sensors for use in the LDRD's third year demonstration on the HAGAR vehicle. GPS would be available intermittently, but capability must be demonstrated while GPS is unavailable. In addition, they would also develop a navigation error analysis tool (NavCov) which could predict navigation errors as a function of arbitrary configurations of component devices.

Sandia technology for missile and bomb navigation was applied to the land navigation problem through the use of a SANDAC flight computer, NovAtel GPS receiver, and a Litton LN200 Inertial Measurement unit (IMU). Inertial Measurement Units (IMU's) typically contain three accelerometers and three gyros, and most contemporary ones are fixed to the vehicle. A navigation solution to determine current "state" (i.e., position, velocity, and attitude) is derived from the IMU instrument measurements. The performance of an IMU depends on the quality of the accelerometers and gyros, where dominant error sources are the instrument bias errors (provided the operating conditions are relatively benign). A good metric for the navigation solution error growth is the time elapsed before the position error is 1 mile. The following table provides generic specifications as a function of cost.

IMU Quality	Bias Errors		Cost ($K)	Time to 1 mile error	Typical Application
	Rate Gyros (degrees/hr)	Accelerometers (milli-g's)			
Navigation Grade	<0.01	<0.05	>60	>1 hr	Aircraft navigation
Tactical Grade	1	0.25	20	~15 min	Missile Navigation
Instrumentation	>500	>10	<5	< 5 min	Stabilization

Reference: Kim, T.J., "Inertial Navigation", viewgraph presentation, 8/2000

Table 1. Inertial Measurement Unit comparison as a function of unit cost

A computer, GPS receiver, and IMU combination provided a baseline inertial navigation system (INS) for measurement acquisition and position estimation (Figure 19) for use on HAGAR. The idea was to have this configuration be the starting point for transitioning to less expensive units and compensating for the loss of accuracy through improved modeling of the accumulated errors and sensor fusion via Kalman filtering.

During this project, two low cost IMU's were tested as possible measuring platforms to which to transition. Unfortunately, they proved to be consistent with instrumentation-grade unit specifications and as such were deemed too inaccurate for land navigation (Appendix B). A thrust of the navigation research in this project was to determine if a set of inexpensive components (tilt sensors, compass, odometers, low-end IMU's) could approach the performance of a missile-grade INS system (~$25-50K). As of this writing, the results have not been encouraging [16].

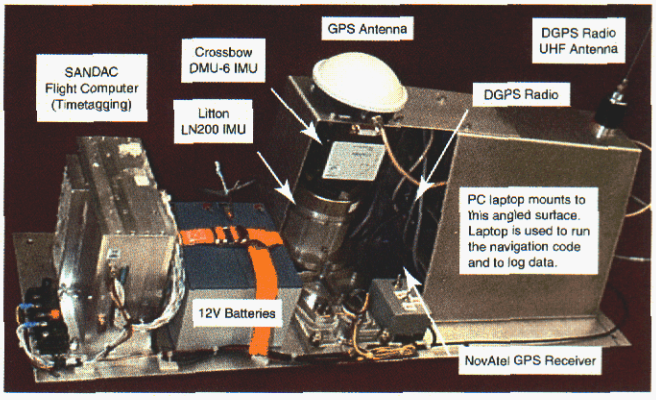
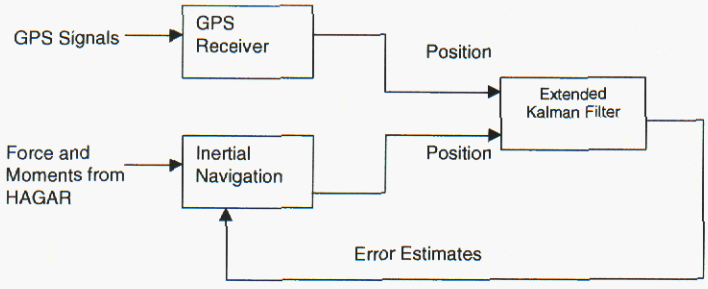


Figure 19. Baseline Navigation Configuration

Most inertial navigation systems are put on vehicles that are constantly moving, predominately flight vehicles. These vehicles can also aid there position by being able to augment computed guidance solutions with terrain-aided techniques which rely on measurements of rapidly changing ground features (due to vehicle speed) to facilitate comparisons with stored maps. Since the robotics application is a comparatively slow speed one, the vehicle will compensate for this lack of area coverage by stopping often to create a known condition on which the sensors can continuously re-calibrate and re-align [15].

The INS system used here subscribes to a closed-loop Kalman Filter configuration in which instrument biases and errors are estimated using GPS inputs and IMU computations. These are fed back to correct IMU parameters during vehicle navigation. A block diagram of this process is shown Figure 20. Filtering can correct for equipment accuracy within limits.



Reference: Jordan, J.D, Dept. 2338, personal communication, Feb 2002

Figure 20. Closed-loop Kalman Filter INS used for HAGAR

It is incumbent upon the analyst to decide how to manage the cost versus accuracy issue for the specified mission. In order to determine navigation error behavior, a ground trajectory that exercises all of the relevant dynamic states of motion measured by the candidate navigation instrumentation is needed. A six-degree-of-freedom trajectory simulator was created to provide vehicle translational and rotational state histories. (In a real scenario, this information would be measured by the IMU). Planar steering and

throttle are the commanded inputs. Figure 21 shows typical simulated position output using a stop-and-go throttle input history. The throttle input is superimposed on the ground track and given a fabricated scale for visualization purposes and to demonstrate coordination with the ground track. Though the ground track is plotted at even time increments the points are unevenly spaced due to the controlled, discontinuous motion in the trajectory. This is done to accommodate a "zero-velocity updating" correction to the estimated position (subsequently explained). The elevation data in Figure 21 is a DTED Level IV model (3 meter post spacing) of Sandia's Robotic Vehicle Range (RVR).

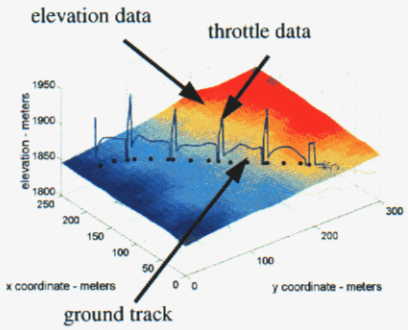


Figure 21. Nominal trajectory for guidance analysis

A known missile INS calibration technique, *zero-velocity updating* (ZVU), was extended for land applications under this effort for improving ground position estimation [15]. This technique employs a stop-and-go operation and the assumption of constant acceleration errors that occur during the motion parts of the trajectory. Reiterating, vehicle motion measurements are processed through a standard set of inertial navigation equations to compute position histories and secondly through a Kalman filter to estimate errors in navigation model constants and the various state histories. These corrections are then used to iteratively update the navigation model (Figure 20). However even missile grade inertial measurement apparatus will cause position estimates to drift substantially and continuously without some updating scheme such as GPS.

A corrective approach is to assume constant acceleration errors are present in the three body-fixed vehicle axes over a motion ("go") segment. These are computed using a coarse finite difference approximation from the velocities at the ends of these segments. These are then double integrated to provide position error corrections (in latitude, longitude, and altitude), which are applied to the initial position estimates.

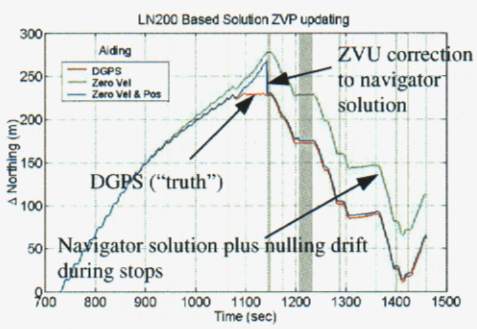


Figure 22. ZVU effects on northing (latitude) component

Figure 22 displays a latitude comparison using the LN-200 Litton inertial measurement unit (IMU). Differential GPS (DGPS) is used as the truth model, but is not used to aid the proposed technique. Note the substantial effect of the correction at 1150 seconds.

Applying this technique with the actual live hardware in a repetitive fashion has maintained better than 15-meter accuracy over 90 minute periods [17].

The development of the navigation error analysis code, *NavCov*, [16] allows one to do tradeoffs on both accuracies and types of position and attitude measurement components used (i.e., IMU's, compasses, odometers, tilt meters, GPS receivers). Borrowing extensively from Sandia missile applications, this tool uses detailed operational specifications of the various hardware components and can be altered to allow one to see isolated (or combined) component characteristic effects on position estimation performance.

A user's manual for NavCov can be found in [19].

Figure 23 shows the position error reduction of LN-200 navigator (Figure 19) in NavCov with GPS updating applied to the vehicle trajectory in Figure 21. The position error variances in three axes have been used as semi-major axis dimensions of a time dependent error "ellipse". Note that the ellipse initially shows more error in the altitude direction and that the use of Kalman filtering "circularizes" this error during the trajectory. Circular position errors imply that you have reached a point where the error reduction process is proceeding uniformly in all directions indicating equal position estimation accuracy in all directions (the ideal condition).

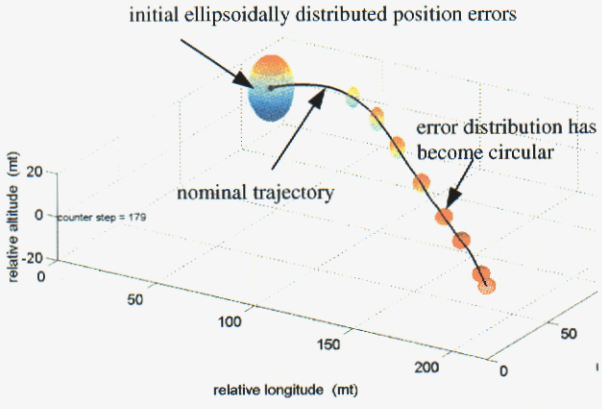


Figure 23. Navigation position error reduction history

Currently, Dept. 2338 is in the process of including the ZVU algorithm in the navigation hardware shown in Figure 19. The navigation Kalman filter was modified to use attitude and velocity information obtained during a zero velocity stop. This has further improved non-GPS aided navigation performance over that of the basic zero velocity + position algorithm. I/O & navigation processor communications protocols have been established and checked out on actual processor boards. The navigation algorithms have also been implemented in this hardware [18].

System Architecture

A Working Group on Modular Mobile Robot Software Architecture was convened to confront the problem of vehicle hardware/software implementations evolving along lines that exhibit high levels of functional interdependence [2]. This situation produces numerous component software functionalities that are intertwined to the point where insertions and deletions have multiple connection points to find and such changes produce severe impacts on stability and reliability in vehicle operation. Software maintenance of such "brittle code" is nightmarish and exceedingly time-consuming. A major goal for an architecture re-work is to provide an environment where functionalities can be packaged such that necessary upgrades and modifications do not cross couple with other areas.
Given that this could be accomplished, a secondary goal is to be able to substitute realistic simulations of hardware components (including the vehicle) for the actual item. This would allow the various functionalities to be isolated and their respective operations tuned safely.

The Working Group recommended:

A sustained funding effort at the 1-2 FTE for at least one full year level to develop and implement a prototype system for a modular architecture, employing both an RTOS and limited use of a distributed processor arrangement. At the minimum, the target mobile robot should have navigation, obstacle/landmark sensing, and route following/vehicle kinematic functions (as well as communication) to integrate. The developed system would

1. benchmark the use of the standardized interfaces between functional blocks to accommodate *significant* modular component connectivity, initialization, and change and replacement within the total system. It is anticipated that this task will generate marked changes in suggested initial compositions of the interfaces.
2. refine the ability of the suggested messaging and communication architecture to handle a representative, multi-function environment and reliably transfer data, and
3. employ a common language (C++ is suggested) among the diverse constituent modules in order to maintain reasonably software consistency. In addition, the working group advocated the implementation of a 32-bit POSIX compliant operating system. This requirement provides support for a multi-threaded (i.e., task) environment that is portable across different operating systems.

Milestone Schedule

This section summarizes the success of the stated milestones listed in The Problem section.

Navigated autonomously over a pre-planned route. In February 2000, a Swarm RATLER (see Mobility) successfully navigated the 1.5-mile distance from Sandia's Intelligent Systems and Robotics Center to the RVR on cleared, dirt roads autonomously via GPS (Figure 24). The route was pre-planned via the recording of waypoints beforehand with the vehicle's GPS receiver for travel along the approximate center of the roadway and guided via the algorithm in the Guidance and Control section. The navigation demonstration was initiated and monitored from a laptop computer running the basestation as depicted in Figure 5. Obstacle avoidance was not an issue for this milestone.

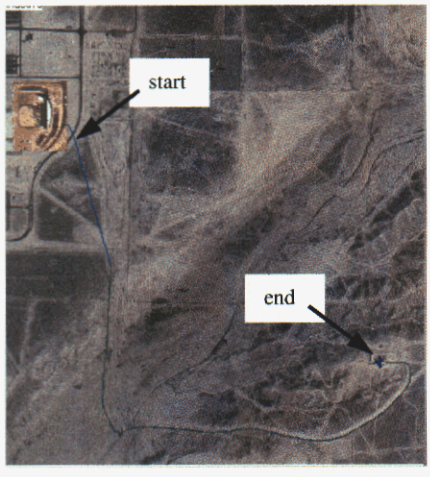


Figure 24. Autonomously navigated route of Swarm RATLER

Navigate autonomously over an established road path that the vehicle plans on-board. In March 2002, the MAT vehicle was able to construct a map of obstacles encountered via bumper switch contact and successfully navigate in a moderately complex environment. Due to the speed of its path planner, it was able to demonstrate that a repetitive "sense and re-path plan" operational approach had potential for solving the general autonomous navigation problem. This capability was achieved at the expense of off-board route planning in order to minimize on-board computational tasks and thus maintain a very low vehicle cost and sufficient computational capacity[12]. This capability could be migrated to the vehicle for a cost increase that would still make this concept attractive to field.

Navigate autonomously off-road over a pre-planned route. Development of inertial navigation hardware and software. The ultimate goal of autonomous navigation in arbitrary environments proved elusive for the HAGAR vehicle due to the lack of a discriminating sensor to determine obstacle "hardness". The ultrasonic sensor arrangement on HAGAR registered excessive false obstacles on the myriad amount of foliage in the test areas. Again, the sense-re-plan paradigm of the MAT vehicle showed the best promise for achieving this milestone. The inclusion of a reliable ranging sensor on MAT would cut down on the wear-and-tear inherent in its current bumper contact-and-re-plan operation.

Zero velocity updating, a missile IMU calibration technique, demonstrated significant potential toward reducing land navigation errors. However, assembling cost-effective inertial measurement navigation hardware (< $10K) that is sufficiently accurate does not appear feasible given the current state-of-the-art.

Lessons Learned

Autonomy for mobile robots is an exciting area of research with huge payoffs for contemporary and future warfighting missions. The following short list of lessons summarizes major "obstacles" (all pun intended) that were encountered and should be addressed in future work.

An autonomous mobile robot is a complex device with numerous cooperating and competing functionalities. Cooperating in the sense that, for example, obstacle sensing must provide information for the path planning and guidance functions to accomplish their tasks. Competing in the sense that many of these functionalities are time-consuming processes that require significant number crunching and computer access. The operational manifestation of these functionalities results in vehicle behavior that is difficult to separate for error debugging. For example, an inaccuracy in path following may be due to shifting satellite reception, poorly tuned guidance parameters, inaccurate navigation solutions, or map errors. Failure to elude obstacles may be due to sensors that "did not see" or guidance measures that "did not compensate".

The following are some lessons/suggestions for future work.

1. The current test facilities at Sandia for analyzing the various functionalities would benefit enormously from the simulation and real-time operating suggestions discussed under System Architecture. These concepts would decouple individual functionalities, thus creating a plug-and-play atmosphere that would clarify effects from and significantly lessen system impact due to upgrades, code and component changes, and tuning,

2. The issue of a sensor to provide a measure of "terrain penetrability" based on vehicle scale is as relevant and elusive now as it was when this project started. Such a development would allow the field of mobile robotics to make a quantum leap in its quest for autonomy and significantly increase the field's penetration into defense world. This development should be pursued with all due haste, and
3. A number of simulations were written in the course of this work, but models for the various functionalities, particularly vehicle mobility and obstacle sensing, were sparsely covered in the literature. Said models have little "hard" validation as versus those describing, for example, flight vehicles. More fundamental theoretical work is needed to provide computational models, which would have direct payoffs for suggestions in the first suggestion.

References

1. Klarer, P.K. [1994], *A Highly Agile Ground Assessment Robot (HAGAR) for Military Battlefield and Support Missions*, SAND 94-0689C, March 1994, **Sandia National Laboratories**
2. Eisler, G. R. et al.," *Preliminary Component and Software Interface Specifications for Input/Output Operations for Navigating Mobile Robot Architectures*", SAND 2002-0815, April 2002
3. Klarer, P.K. [1993], *Recent Developments in the Robotic All Terrain Lunar Exploration Rover (RATLER) Program*, SAND93-1760C, August 93, **Sandia National Laboratories**
4. Harrington, J.J., *Development of a Self-Navigating Mobile Interior Robot for Application as a Security Guard-Sentry,* SAND86-0653J, 1986
5. Harrington, J.J., *SIR-1: an autonomous mobile sentry robot,* SAND87-1128
6. Feddema, J., Lewis, C., Klarer, P. [1999], *Control of Multiple Robotic Sentry Vehicles*, **Proceedings of the SPIE , Unmanned Ground Vehicle Technology**, Orlando, April
7. Massa Products Corporation, Hingham, MA, www.massa.com
8. Nilsson, Nils.J. Problem-solving Methods in Artificial Intelligence, **McGraw-Hill**, Inc, 1971
9. Stentz, A. [1994], *Optimal and Efficient Path Planning for Partially-Known Environments,* **IEEE International Conference on Robotics & Automation,** May 1994
10. Lewis, C., personal communication, Sandia National Laboratories, Department 15211, Albuquerque, June 2002
11. Klarer, P. R., personal communication, Department 15252, Sandia National Laboratories, Albuquerque, June 2002
12. Harrington, J.J., personal communication, Department 15252, Sandia National Laboratories, Albuquerque, June 2002
13. The MathWorks Inc., **MATLAB - The language of Technical Computing**, Version 6, Nov 2000, Natick, MA, www.mathworks.com
14. Bradley, J.D., "AutoNav - Inertial Navigation Task – Statement of Work, July 2001, Sandia National Laboratories
15. Kim, T.J., "Inertial Navigation", viewgraph presentation, Department 2338, Sandia National Laboratories, Albuquerque, August 2000

16. Bradley, J. D., personal communication, Sandia National Laboratories, Department 2338, Albuquerque, November 2001

17. Jordan, J.D., "Zero Velocity + Position Updating", viewgraph presentation, Department 2338, Sandia National Laboratories, Albuquerque, February 2002

18. Bradley, J.D., "March/April 02 Status Report - AutoNav & DistAutoNav Navigation Tasks", memo to R. Eisler, May 2, 2002, Sandia National Laboratories

19. Eisler, G. R., "Distributed Autonomous Navigation – An LDRD Final Report", Sandia National Laboratories, Albuquerque, to be published in September 2002.

20. Dohrmann, C. R., personal communication, Sandia National Laboratories, Department 9124, Albuquerque, August 2001,

Appendix A: Navigating Mobile Robot Architecture Glossary

Attitude
vehicle rotation angles defined by pitch, yaw, and roll angles with respect to a world inertial frame what we have in abundance

Autonomous Navigation
point-to-point vehicle traversals without human intervention – the Holy Grail!!

Bearing
azimuth angle with respect to an arbitrary azimuth reference angle, typically either a compass angle (for world referenced bearings) or vehicle longitudinal centerline (for local-vehicle referenced bearings)

Command
instruction sent to vehicle to initialize variable and components or initiate activity

Communications – interfacing data flows from onboard functional modules or off-board to the communications backbone (via radio, hardwire or other transfer mechanisms)

Control
a high level computing entity that deduces wheel speeds (or other propulsive entities) using inputs from route planning, navigation, and other sensing modules

Dead Reckoning
a method of navigation based on odometry and vehicle compass bearing angles

DEM
Digital Elevation Model – matrix of elevations as a function of latitude and longitude

DOS
Microsoft's Disk Operating System

Encoders
devices that measure wheel (or joint) rotations or positions

Functional Module
a software routine or self-contained module for accomplishing a generic vehicle function (route planning, navigation, obstacle/landmark sensing, etc.)

Goal
desired vehicle state at a specified point in the traverse.

HAGAR
High Agility Ground Assessment Robot

Inertial Measurement Unit (IMU)
a device which typically measures rates that describe changes of the six degrees of freedom (3 rotation, 3 translation) of a moving entity with respect to a local level (inertial) system. Integration of these measurements can produce position and attitude.

Inertial Navigation System (INS)
combination of sensors and algorithms that provides navigation solutions for position, velocity, and acceleration

Interface

means of connecting functional modules in either software or hardware

MAT

Minimalist Autonomous Testbed

Navigation

a functional module that combines sensor measurements and algorithms to compute a best estimate for current vehicle position (also may include attitude and velocity)

Obstacle

a physical impediment to planned ground trajectory travel, can be above or below the average surface and may not necessarily be physically connected to the surface (overhanging branch or structure)

Obstacle Sensing

an asynchronously operating functional module that combines sensor measurements and algorithms to identify position and character of obstacles relative to the vehicle position

Odometry

measuring accumulated wheel rotations or position, can be absolute or relative measurement

Path Planning

a functional module that generates paths(list of intermediate waypoints) between the starting point and the destination that optimizes or satisfies the objectives, like obstacle avoidance, minimal energy usage, line of sight traversal etc. The Path Planning module typically generates a path segment that is free from collisions, maintains line of sight for communications and minimizes the expected energy used.

Plug-and-Play

the ability to modify an overall vehicle architecture through the addition, replacement, or elimination of entire functional modules via standardized interfaces.

POSIX

Portable Operating System Interface

Range

distance from the vehicle to a goal, obstacle, or landmark

Real-Time Operating System (RTOS)

a multi-tasking environment that coordinates task operation and timing via assigned priorities and interrupts

Route Planning

path planning. A functional module that incorporates mission-planning goals to define a specific point-to-point trajectory solution in terms of vehicle-measured or computed quantities, such as waypoints.

Route Following

a functional module that generates wheel velocity set points to follow a given path or navigational objective. The Route Following module uses the vehicle's current attitude, position, and the most recent map data to generate wheel velocities to achieve the current navigational objective. This, and a vehicle dynamics module could be passed to a RK45 numerical integration routine for simulating the vehicle.

RSTA

reconnaissance, surveillance, and target acquisition .

Servo-level Control

high bandwidth operation to assure that low-level component outputs (i.e., wheel motor speeds) are slaved to desired values.

Single threaded

Use of a single continuously running process to operate all vehicle functions in a serial, non-prioritized fashion.

Standardization

enforcing design specifications that must be met to interface components or software

Thread

a computational task or process

Trajectory

a list of waypoints.

Trajectory Control

synonymous with Route Following.

Vehicle Dynamics

plant dynamics

Vehicle Kinematics

a functional module that combines Navigation data with Velocity Vector solutions to generate wheel speeds (or other propulsive outputs)

Vehicle state

the vector of quantities that describes the condition of the vehicle in terms of its static and dynamic variables with respect to the world inertial coordinate frame and other designated "internal health" variables

Waypoints

ordered pairs of latitude and longitude that comprise a trajectory for the vehicle to follow

Zero Velocity Updating (ZVU)

Missile INS calibration technique adapted in this study for land vehicles

Appendix B: Scoring the Microbotics μINS against NovAtel differential GPS.

The following test and results of a low-end IMU system were provided by Robert Morris, Dept. 2338, rjmorri@sandia.gov.

Physical Arrangement of the roaming systems: The μINS equipment and the NovAtel antennas were fastened atop a specially equipped van as shown in figure 1. The μINS IMU housing was secured between the parallel runners that supported the GPS pinwheel antenna that would collect the data used to post-process the differential position of the van and the μINS. The μINS antenna was carefully located as shown to the right of the μINS IMU housing. Two additional antennas (not shown) were positioned inline parallel to this arrangement. These antennas were dedicated to a second NovAtel system for obtaining stand-alone attitude, position, and velocity measurements that would be beneficial in extending the analysis.

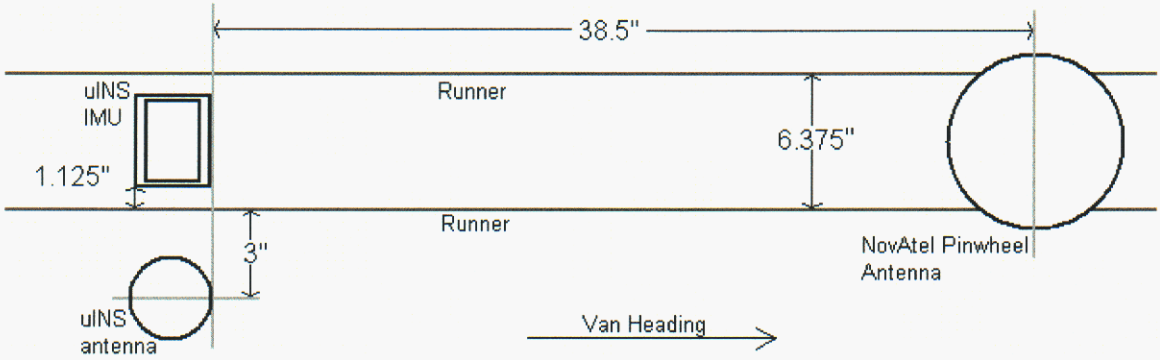


figure 1 – Physical arrangement of the μINS Navigation system and the NovAtel grading system

Execution: Multiple tests were performed and analyzed; however, only one set of data will be discussed in detail. Another will be mentioned in the Summary section of this report.

The μINS and the NovAtel systems were each powered on and allowed time to derive a solution. A software graphical interface is provided with the μINS as with the NovAtels. Once all systems, the μINS, two roaming NovAtels, and the base NovAtel, reached their solutions, data logging was enabled on each.

The data collection for the μINS was broken into segments to prevent corruption losses. The first segment consisted of the travel from the base location (building 891) to the testing pad. After reaching the testing pad, the second segment consisted of a series of motions: large circles, small circles, figure eights, and forward and reverse lines. The

third segment was similar to the second except that it was preceded by a power reset of the μINS. It was somewhat apparent from the μINS graphical interface that the μINS had lost its solution in the second segment. The μINS was allowed to regain its solution before proceeding. The forth segment consisted of a power reset, the deliberate loss of GPS, a few simple motions, the resurrection of GPS, and the return trip to the base. All motion aside from the travel between the test pad and the base was capped at 15 – 20 mph, and systems were continually monitored for proper operation.

Test Results:

Segment 1: At first glance of figure 1.1, The μINS appears to follow the NovAtel fairly well. Figure 1.2 indicates that the μINS assures that the measurements are in fact reliable for the duration of this segment. A receiver mode of four indicates a good 3D solution, and the receiver appears to be tracking a substantial number of satellites. Figure 1.3 provides a reference to NovAtel diagnostics, and Figure 1.4 clearly identifies blatant error in the μINS system. The two systems are synchronized using GPS time, and from thus, the differences are calculated.

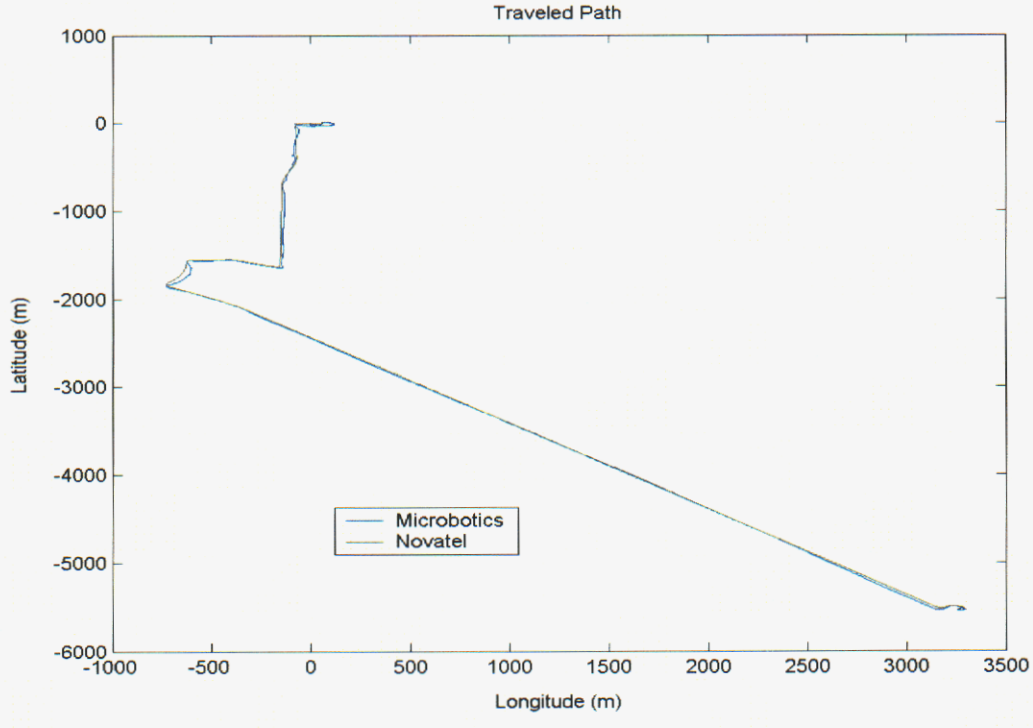


figure 1.1 - the path taken from the base to the testing pad as recorded by μINS and NovAtel differential GPS. Speeds up to 50 mph were obtained.

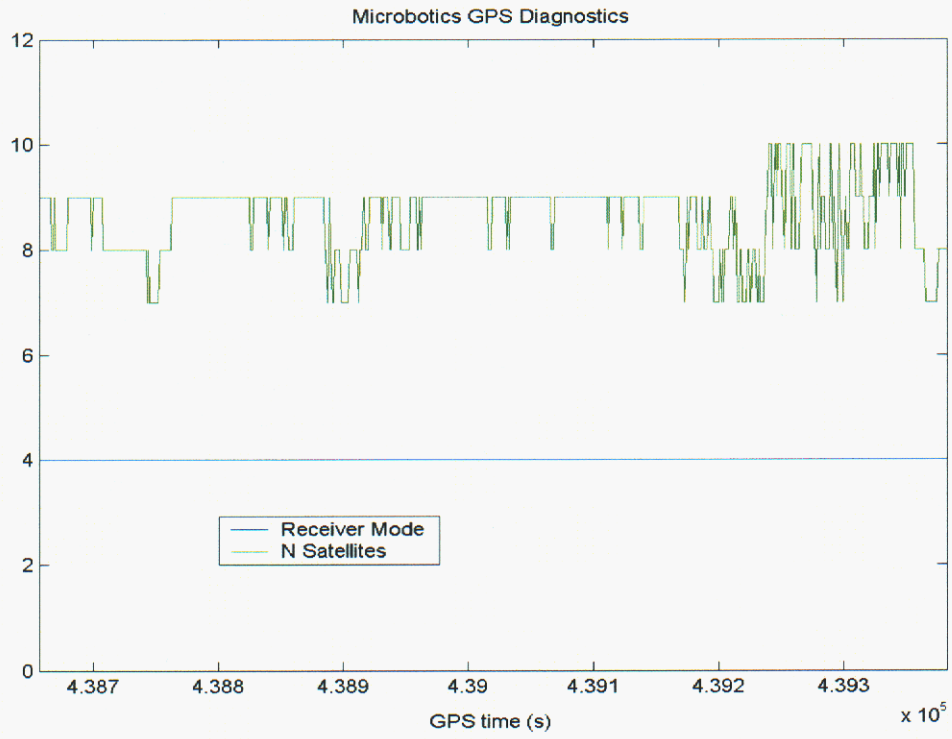


figure 1.2 – status of μINS GPS for the duration of segment 1

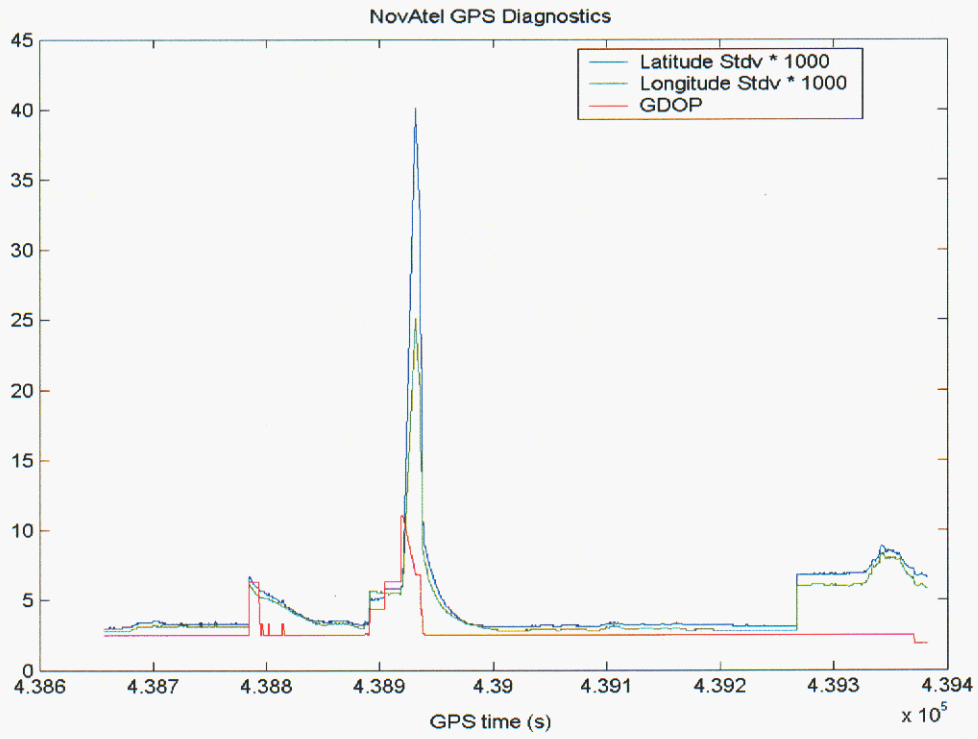


figure 1.3 – Standard Deviations of NovAtel's position solution for the duration of segment 2

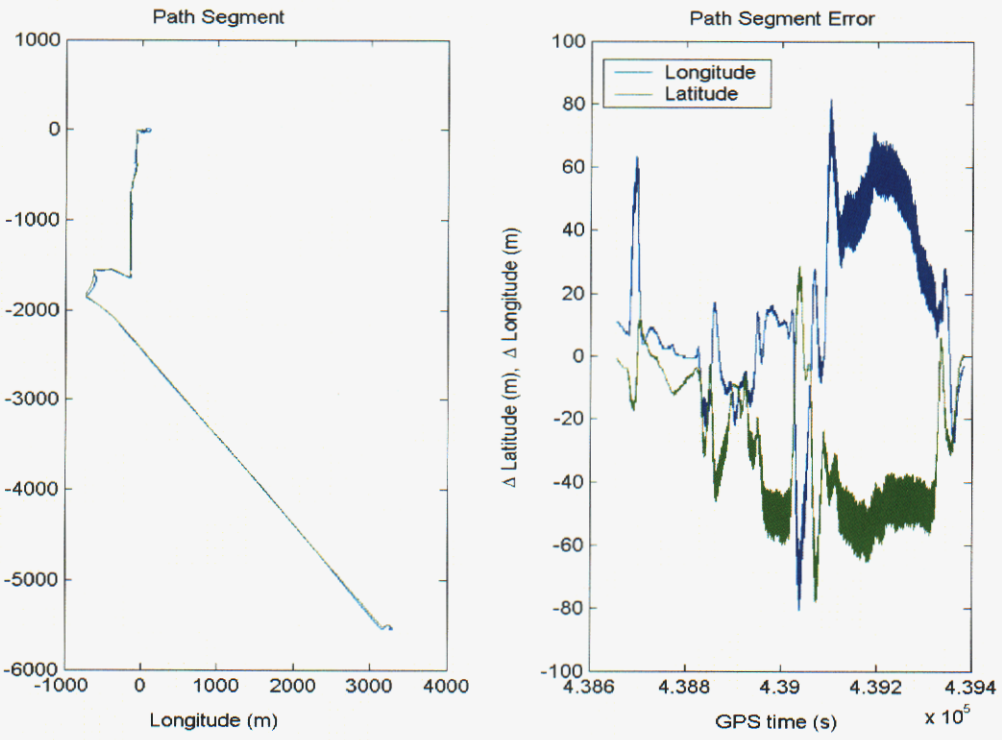


figure 1.4 – The difference between μINS Latitude & Longitude and differential Latitude & Longitude in meters

Segment 2: Figure 2.1 hides nothing in regards to μINS error. Again, figure 2.2 indicates that μINS assumes a quality solution status for the duration of the segment. Figures 2.4, 2.5, and 2.6 quantify the error for each motion that was attempted and successfully recorded.

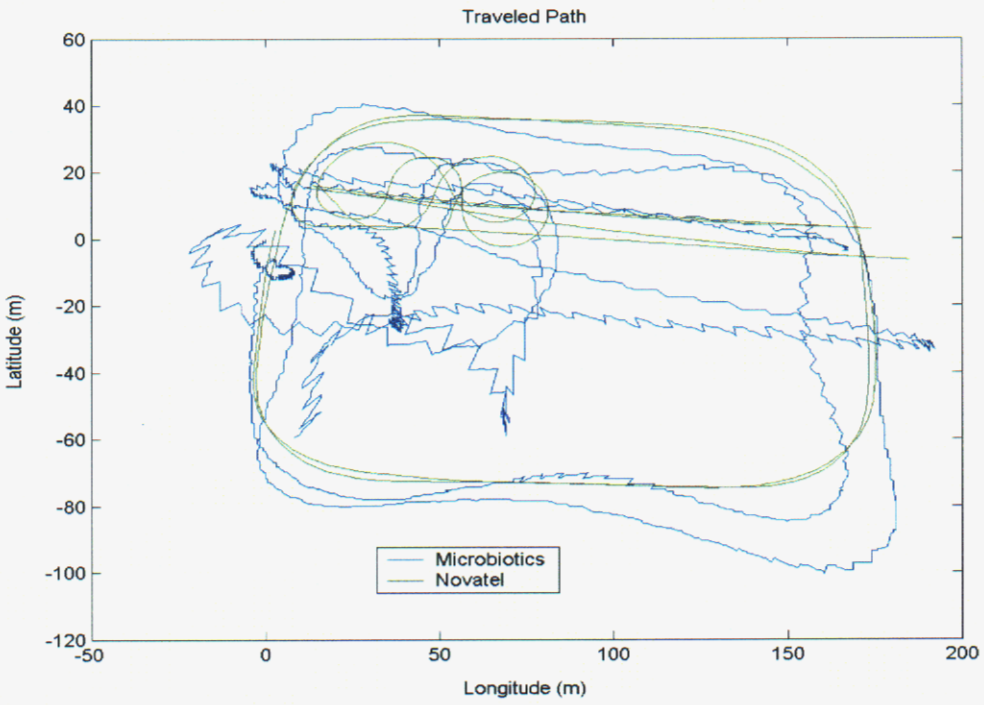


figure 2.1 - the path taken for the first segment on the testing pad as recorded by μINS and NovAtel differential GPS.

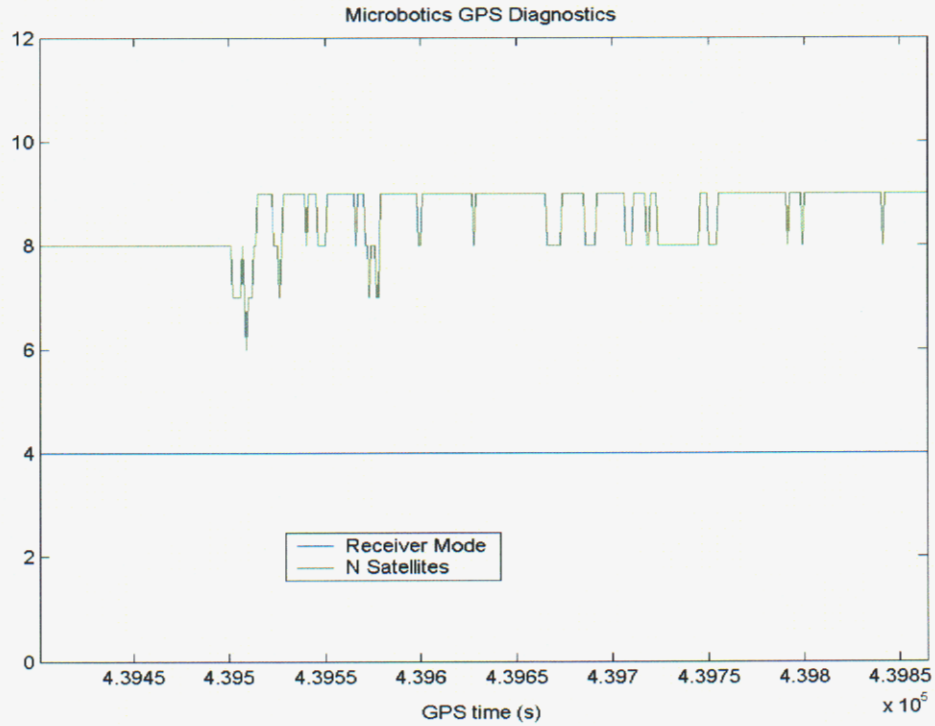


figure 2.2 – status of μINS GPS for the duration of segment 2

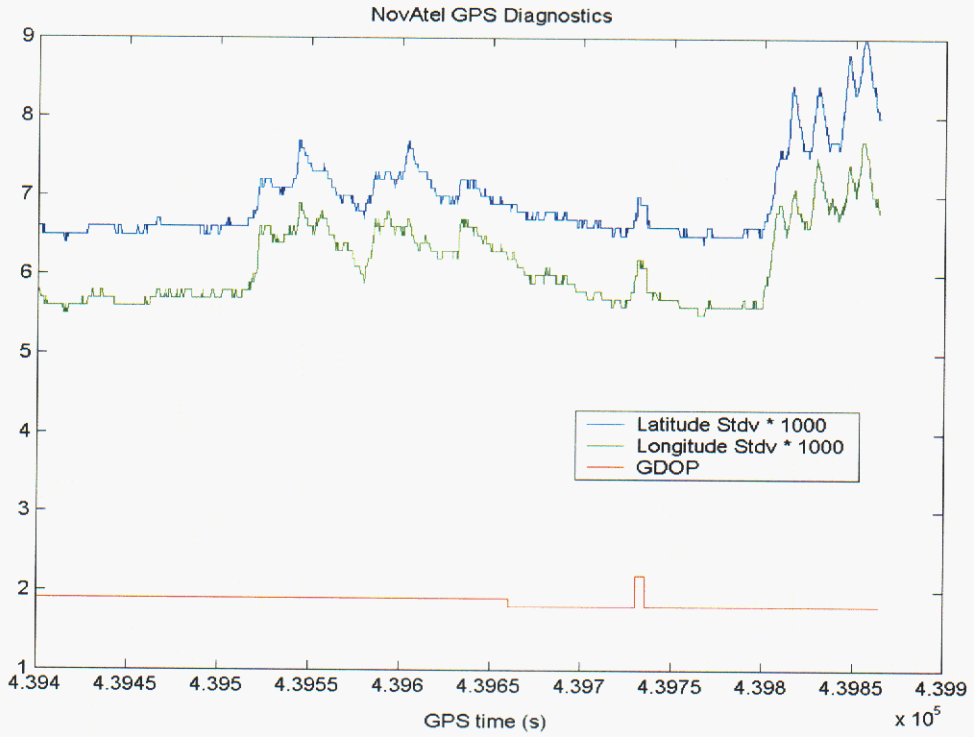


figure 2.3 – Standard Deviations of NovAtel's position solution for the duration of segment 2

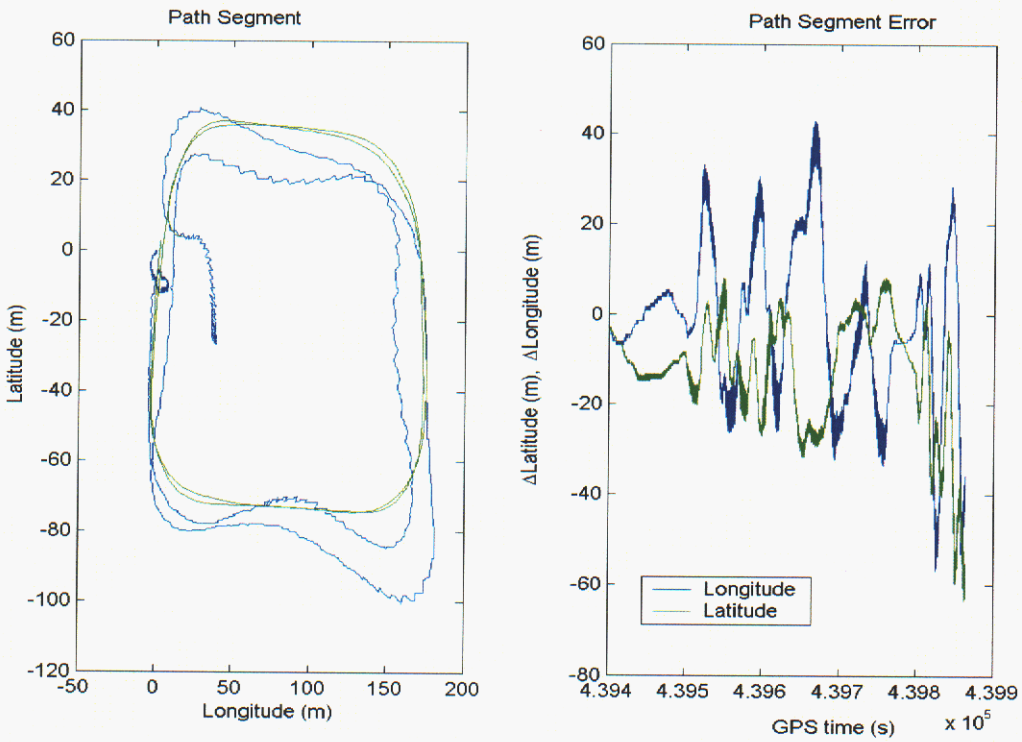


figure 2.4 – The difference between μINS Latitude & Longitude and differential Latitude & Longitude in meters for two large circles

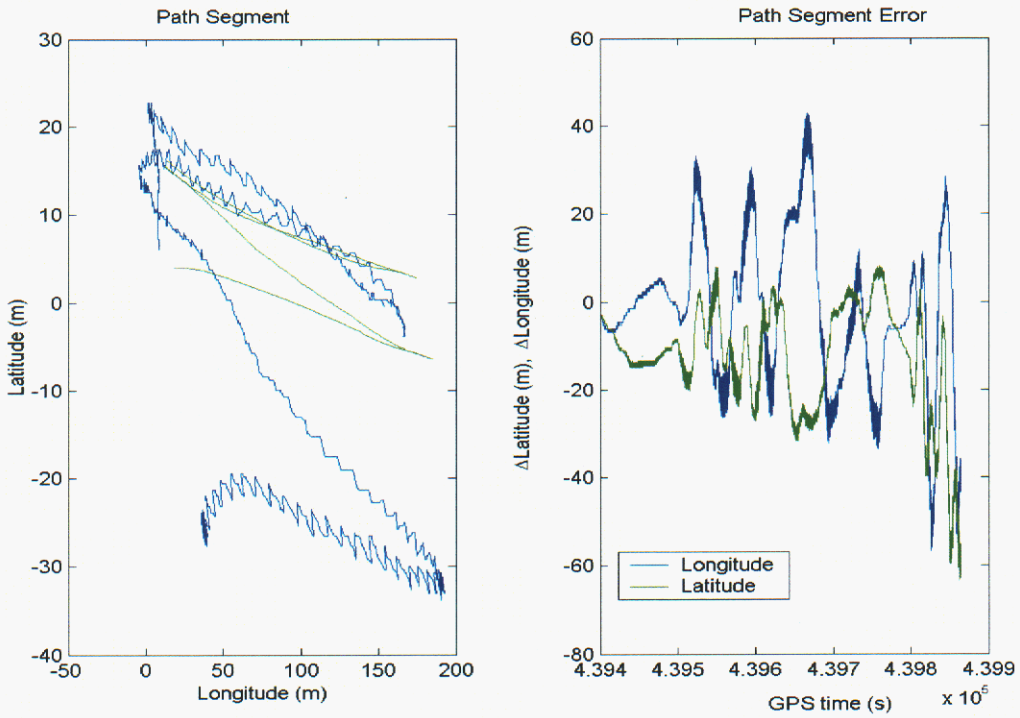


figure 2.5 – The difference between μINS Latitude & Longitude and differential Latitude & Longitude in meters for forward and reverse travel

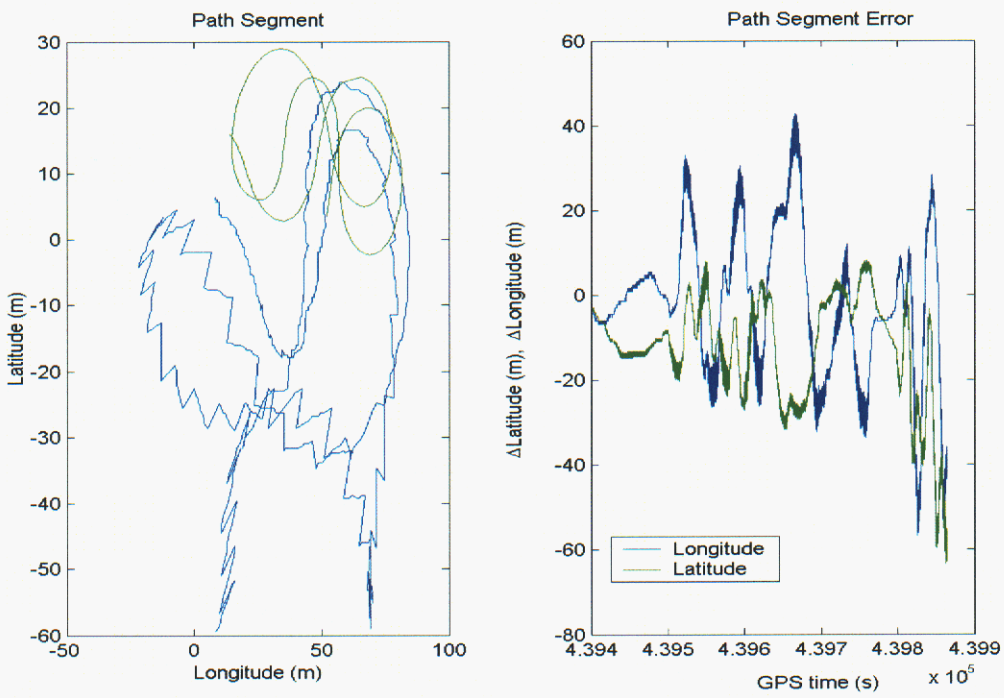


figure 2.6 – The difference between μINS Latitude & Longitude and differential Latitude & Longitude in meters for two figure eights

Summary:

The testing pad is of large rectangular asphalt construction and is clear of power lines and overhead coverage.

NovAtel differential GPS for roaming subjects is by professional opinion accurate to within 20 cm.

Midway through segment 2, the data file was corrupted. Segments 3 and 4 failed entirely. Although the graphical interface indicated that the solution was solid, GPS was tracking, and data was logging for the remainder of this test, valid data was not logged again following the first power reset. However, the files pertaining to these segments were not corrupt.

Any intentions to take the systems out again to try to recapture the lost data have been thwarted by the poor performance of the µINS.

In other tests, the µINS has failed to produce reliable results all the while the system claims to be recording a valid solution. For example, figure 3 describes an earlier test day.

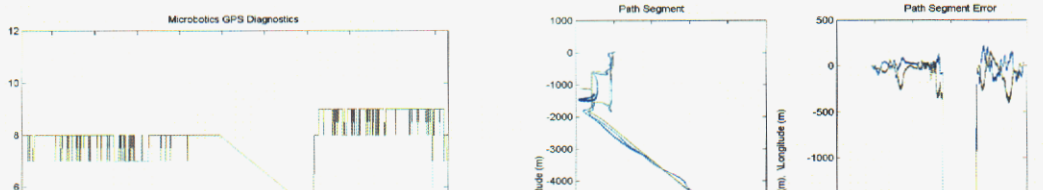


figure 3 - µINS receiver status and the erred data that was recorded

In this test, the first segment of the test failed due to equipment operation error. The second segment was started after the error was corrected. In the first segment, GPS had been removed for a time possibly explaining the distorted results found in segment 2. The point is that the µINS claims a dependable solution when it is obviously incorrect.

Documented Test Procedure

Segment 1 –

The path from the base to the pad was logged.

Segment 2 –

GPS Time	Motion
439490	2 large circles around pad, CCW
439650	Forward, Reverse, Forward, Reverse for the length of the pad
439790	2 figure eights (1 CCW, 1 CW)
439900	Forward, Reverse for the length of the pad with jerkiness
439990	2 tight circles

Segment 3 –

After power reset, solution was obtained at 440178 with 5 satellites in view

GPS Time	Motion
440280	2 large circles around pad, CCW
440430	Forward, Reverse, Forward, Reverse for the length of the pad
440560	2 figure eights (1 CCW, 1 CW)
440760	Forward, Reverse for the length of the pad with jerkiness
440840	2 tight circles

Segment 4 –

After power reset, solution was obtained at 441100. At 441260, 7 satellites were in sight with steady GPS updates – good 3D solution. GPS was removed at this time. Steady GPS updating was eliminated. The GUI still indicated 7 satellites and good 3D solution. Simple motions were driven.

At 441510, GPS was reinserted. Steady GPS updates resumed and system status remained at high level. The return to the base was allowed to be logged.

Appendix C: Derivation of nominal vehicle trajectory model for navigation analysis

The trajectory analysis code provides a pseudo-six-degree-of-freedom motion model by combining point mass equations for translational state rates with a finite-differencing of the local terrain slope as a function of vehicle speed to provide attitude state rates.

The following model assumptions have been made:

1. Motion is modeled in a right-hand-coordinate system
2. There is no vehicle motion normal to local surface.
3. Controlled motion of the vehicle is along its longitudinal ($\vec{e}_{X_B}$) axis (see Figure C1)
4. The vehicle can only move along its lateral axis if the lateral gravity component is greater than the lateral frictional force exerted by the vehicle-terrain interaction.
5. The vehicle is controlled via a throttle acceleration input along the longitudinal axis and an angular rate steering input about an axis normal to the local tangent-to-the-surface plane.

A terrain sample and the world and vehicle axis systems are depicted in Figure C1.

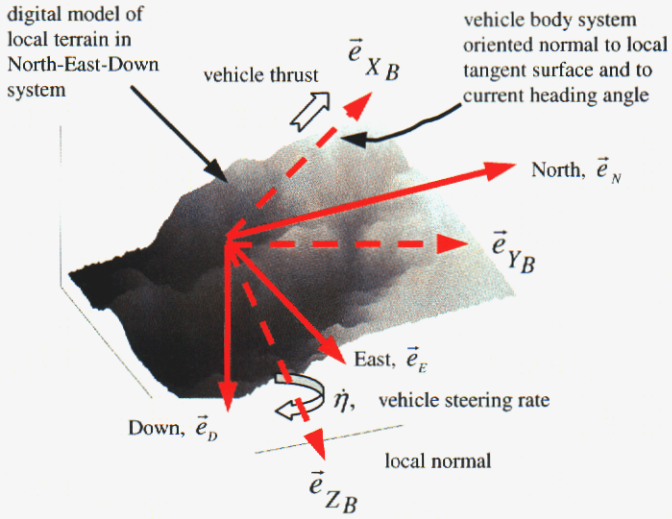


Figure C1. Terrain axis systems for vehicle trajectory analysis

To resolve activity in the vehicle frame to establish position in the world frame we start by developing a direction cosine transformation between the world $\vec{e}_N, \vec{e}_E, \vec{e}_D$ system and a local system ($\vec{e}_x, \vec{e}_y, \vec{e}_z$) tangent to the terrain. To facilitate this, we construct the geometry in Figure C2.

After Dohrmann [20], the equation of the plane is $Z(X,Y) = -aX + bY$ where $\frac{\partial Z(X,Y)}{\partial X} = -a(X,Y)$, $\frac{\partial Z(X,Y)}{\partial Y} = b(X,Y)$ and $a(X,Y)$, $b(X,Y)$ are computed via curve fits of the terrain elevation data.

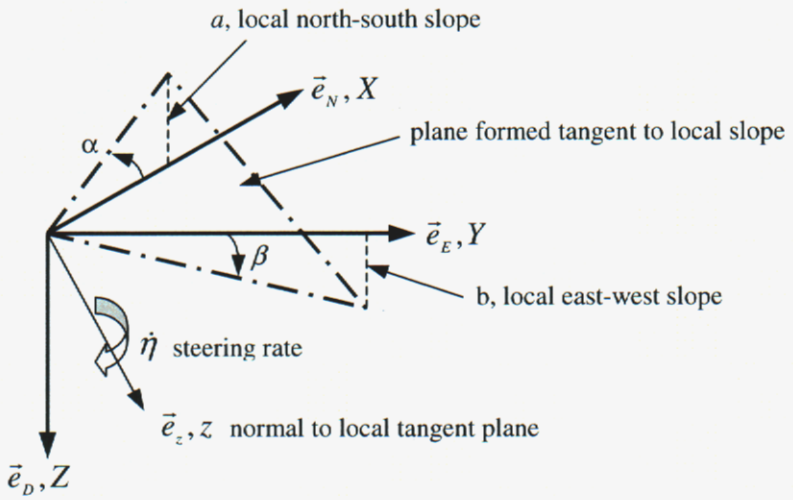


Figure C2. Constructing the local tangent plane

The normal to the tangent plane is given by $(\vec{e}_N - a\vec{e}_D)\times(\vec{e}_E + b\vec{e}_D) = \vec{e}_D + a\vec{e}_N - b\vec{e}_E$. Thus the unit tangent plane normal is $\vec{e}_z = \frac{a\vec{e}_N - b\vec{e}_E + \vec{e}_D}{\sqrt{1+a^2+b^2}}$. If the tangent plane axis, $\vec{e}_x = \frac{\vec{e}_N - a\vec{e}_D}{\sqrt{1+a^2}}$, then the third tangent plane axis $\vec{e}_y = \vec{e}_z \times \vec{e}_x$ or $\vec{e}_y = \frac{ab}{d_1 d_2}\vec{e}_N + \frac{1+a^2}{d_1 d_2}\vec{e}_E + \frac{b}{d_1 d_2}\vec{e}_D$, where $d_1 = \sqrt{1+a^2+b^2}$ and $d_2 = \sqrt{1+a^2}$.

From this the direction cosine transformation from the world frame to local tangent is given by

$$\begin{bmatrix} \vec{e}_x \\ \vec{e}_y \\ \vec{e}_z \end{bmatrix} = [T_{World\_to\_local}] \begin{bmatrix} \vec{e}_N \\ \vec{e}_E \\ \vec{e}_D \end{bmatrix}, \quad \text{where } T_{World\ to\ local} = \begin{bmatrix} \frac{1}{d_1} & 0 & -\frac{a}{d_1} \\ \frac{ab}{d_1 d_2} & \frac{1+a^2}{d_1 d_2} & \frac{b}{d_1 d_2} \\ \frac{a}{d_2} & -\frac{b}{d_2} & \frac{1}{d_2} \end{bmatrix}$$

A final rotation is provided for vehicle steering to complete the transformation to the vehicle frame ($\vec{e}_{X_s}, \vec{e}_{Y_s}, \vec{e}_{Z_s}$). If the vehicle changes its local heading in the tangent plane by an angle, η, the complete transformation to the vehicle frame is given by

$$T_{World\ to\ veh} = \begin{bmatrix} c\eta & s\eta & 0 \\ -s\eta & c\eta & 0 \\ 0 & 0 & 1 \end{bmatrix} T_{World\ to\ local} = \begin{bmatrix} \frac{c\eta}{d_1} + \frac{(ab)s\eta}{d_1 d_2} & \frac{(1+a^2)s\eta}{d_1 d_2} & -\frac{(a)c\eta}{d_1} + \frac{(b)s\eta}{d_1 d_2} \\ -\frac{s\eta}{d_1} + \frac{(ab)c\eta}{d_1 d_2} & \frac{(1+a^2)c\eta}{d_1 d_2} & \frac{(a)s\eta}{d_1} + \frac{(b)c\eta}{d_1 d_2} \\ \frac{a}{d_2} & -\frac{b}{d_2} & \frac{1}{d_2} \end{bmatrix}$$

where $c\eta = \cos(\eta), s\eta = \sin(\eta)$.

The translational vehicle motion is expressed in accordance with Newton's law as $\vec{F} = m\frac{d\vec{V}}{dt} = m\left[\frac{\partial \vec{V}}{dt} + \vec{\omega} \times \vec{V}\right]$ where $\vec{F}$ is the sum of the external forces on the vehicle, m is the vehicle mass, $\vec{V}$ is the velocity vector, $\vec{\omega}$ is the vehicle angular velocity vector, and t is time. The above vectors are expressed as

$\vec{\omega} = p\vec{e}_{X_B} + q\vec{e}_{Y_B} + r\vec{e}_{Z_B}$, $\vec{V} = u\vec{e}_{X_B} + v\vec{e}_{Y_B} + w\vec{e}_{Z_B}$ and,

$\vec{F} = mg\vec{e}_D + \left[\frac{\tau_{wheel}}{r_{wheel}} - R * sign(u)\right]\vec{e}_{xB} - \mu m g_{ZB}\vec{e}_{YB}$, where p,q,r are the vehicle roll, pitch, and yaw rates, $u,v,w,$ are the translational vehicle velocity components in the vehicle frame, g is the gravity constant, μ is the friction coefficient, τ_{wheel}, r_{wheel} are the vehicle torque and wheel radius, R is the soil resistance, and

$$g_{ZB} = z - component\ of \left[T_{World\ to\ veh}\right]\begin{bmatrix} 0 \\ 0 \\ g \end{bmatrix}.$$

Given the initial assumptions, the differential equations of translational motion along the ground are given by the individual components of the vehicle frame velocity rates as:

$$\dot{u} = g_{xB} + \frac{\tau_{wheel}}{m r_{wheel}} - \frac{R sign(u)}{m} + vr$$

$$\dot{v} = g_{YB} - \mu g_{ZB} + ur$$

$$\dot{w} = 0$$

Unless the vehicle is slipping sideways, $g_{Y_B} > \mu g_{Z_B}$, *then* $\dot{v} = 0$ as well. Throttle input, $\tau_{wheel}(t) = k_V(V_{cruise} - V(t))$, where k_V is a proportional velocity gain and V_{cruise} is a designated velocity to achieve. The equations for vehicle frame velocities, u, v, and w, are integrated to provide vehicle frame coordinates, X_B, Y_B, Z_B, which are resolved via $\left[T_{World\ to\ veh}\right]^{-1}$ to yield world coordinates X,Y,Z.

The rotational behavior is provided via differential equations for a and b. For this model, this was accomplished using finite differences where $\dot{a}(t_i, X, Y) = \frac{a(t_i, X, Y) - a(t_{i-1}, X, Y)}{t_i - t_{i-1}}$ at time, $t = t_i$. In the event that $\dot{a}(t_i)$ was approaching zero, a second approximation for

$\dot{a}(t_i)$ was developed according to $\dot{a} = \frac{d}{dt}(a) = -\frac{d}{dt}\left(\frac{\partial Z}{\partial X}\right) = -\frac{\partial^2 Z}{\partial X^2}\frac{dX}{dt}$ where the 2nd derivative is determined via 2nd order fits to the terrain data and the last term is acquired by resolving the vehicle velocity into the world frame for the North component. A similar approach is used to acquire $\dot{b}(t_i)$. Since $a(t), b(t)$ are tangents of angles, they can be converted to terrain angles α, β via the form $\dot{a} = \frac{d}{dt}(\tan\alpha) = \dot{\alpha}\sec^2\alpha$. Substituting this above yields the terrain angular equations

$$\dot{\alpha} = -\cos^2\alpha\frac{\partial^2 Z}{\partial X^2}\frac{dX}{dt}, \quad \dot{\beta} = \cos^2\beta\frac{\partial^2 Z}{\partial Y^2}\frac{dY}{dt}$$

for the north-south and east-west angular slope changes where $\tan\alpha = a$, $\tan\beta = b$.

The steering angle, η, is computed by integrating the equation, $\dot{\eta}(t) = k_\eta \Delta\nu$, where k_η is a proportional steering gain and $\Delta\nu = \tan^{-1}\left(\frac{Y_{gaol} - Y(t)}{X_{goal} - X(t)}\right)$ is the heading change from current vehicle heading to that of the goal.

The five differential equations are integrated with a fixed-step, Runge-Kutta algorithm to determine the vehicle motion. Since $\dot{\alpha}, \dot{\beta}$ are referenced to the world axes, they are resolved into the vehicle frame via $T_{World\ to\ veh}$ and summed with the steering rate to generate approximations to the vehicle angular velocity rates, *p*, *q*, and *r* according to

$$\begin{bmatrix}\dot{p}\\ \dot{q}\\ \dot{r}\end{bmatrix} = \left[T_{World\ to\ veh}\right]\begin{bmatrix}\dot{\beta}\\ \dot{\alpha}\\ 0\end{bmatrix} + \begin{bmatrix}0\\ 0\\ \dot{\eta}\end{bmatrix}$$

A set of Euler angles, $\phi - roll$, $\theta - pitch$, $\psi - yaw$, that would rotate the vehicle sequentially from the world to local vehicle frame, was fabricated according to a solution for a standard $\psi - \theta - \phi$ sequence using the $T_{World\ to\ veh}$ direction cosine entries.

Distribution

1	MS 0501	Jeff Bradley, 2338
2	MS 0973	G. Richard Eisler, 5742
1	MS 1003	Ken Jensen, 15212
1	MS 1003	Denise Padilla, 15212
1	MS 1003	Maritza Muguira, 15212
1	MS 1003	David Wilson, 15211
1	MS 1125	Keith Miller, 15252
1	MS 1125	Paul Klarer, 15252
1	MS 1125	Alex Maish, 1525
1	MS 9018	Central Technical Files, 8945-1
2	MS 0899	Technical Library, 9616
1	MS 0612	Review & Approval Desk, 9612
1	MS 0188	LDRD Office, 1030