A Robust Approach to High-Speed Navigation for Unrehearsed Desert Terrain

Chris Urmson, Charlie Ragusa, and David Ray
Science Applications International Corporation
San Diego, California 92121

Joshua Anhalt, Daniel Bartz, Tugrul Galatali, Alexander Gutierrez, Josh Johnston, Sam Harbaugh, Hiroki "Yu" Kato, William Messner, Nick Miller, Kevin Peterson, Bryon Smith, Jarrod Snider, Spencer Spiker, Jason Ziglar, and William "Red" Whittaker
Carnegie Mellon University
Pittsburgh, Pennsylvania 15213

Michael Clark
Harris Corporation,
Melbourne, Florida 32902

Phillip Koon and Aaron Mosher
Boeing Corporation
Huntsville, Alabama 35824

Josh Struble
Caterpillar Inc.
Peoria, Illinois 61656

Received 22 December 2005; accepted 30 April 2006

This article presents a robust approach to navigating at high speed across desert terrain. A central theme of this approach is the combination of simple ideas and components to build a capable and robust system. A pair of robots were developed, which completed a 212 km Grand Challenge desert race in approximately 7 h. A pathcentric navigation system uses a combination of LIDAR and RADAR based perception sensors to traverse trails and avoid obstacles at speeds up to 15 m/s. The onboard navigation system leverages a

Journal of Field Robotics 23(8), 467–508 (2006)
Published online in Wiley InterScience (www.interscience.wiley.com). • DOI: 10.1002/rob.20126

human-based preplanning system to improve reliability and robustness. The robots have been extensively tested, traversing over 3500 km of desert trails prior to completing the challenge. This article describes the mechanisms, algorithms, and testing methods used to achieve this performance.

1. INTRODUCTION

Autonomously navigating at high speeds for long distances necessitates a robust and capable robot. While there have been numerous examples of highly capable autonomously navigating robots (Kelly & Stentz, 1997; Coombs, Lacaze, Legowik & Murphy, 2002; Wettergreen et al., 2005) none of them have been able to drive challenging desert roads and trails at high speeds for multiple hours (Urmson et al., 2004). Achieving this combination of robotic skill and stamina was a goal of the DARPA Grand Challenge. In meeting the Grand Challenge, two principles emerged as the keys to robustness and success: Keep the components simple and test as frequently and as aggressively as possible. Sandstorm and H1ghlander (Figure 1) are the embodiment of this strategy.

In simplifying the overall robotic systems, a balanced approach using mechanical and software solutions was adopted to take advantage of existing technologies, where possible. For example, the selection of the high mobility multipurpose wheeled vehicle (HMMWV) and Hummer H1 chassis instead of a more readily available sports utility vehicle (SUV) chassis meant that the onboard navigation software need not be as sensitive to terrain features (i.e., an H1's ground clearance is much larger than that of a conventional SUV and so the perception system can ignore rocks that are irrelevant to the H1 but would cause significant damage to a smaller vehicle).

Similarly, human input during a preplanning process is used to reduce the complexity of the onboard navigation system. While the Grand Challenge precluded any intervention while robots were operating on a course, the 2 h before the challenge could be leveraged with human input to increase the likeliness of success. During this time, human editors worked to modify the route to account for dangerous terrain that might be difficult for a robot to detect in real time. The process provided enough information for pre-emptive action to be taken to avoid some dangerous situations and helped reduce the complexity of the onboard navigation system. By balancing simplicity with the necessary complexity required to complete the challenge, a robust system was developed.

The development process consisted of short development cycles interleaved with periods of intensive field testing. While this took a toll on both personnel and equipment, it enabled the discovery of problems and weaknesses with both implementations and ideas simultaneously.

1.1. Format of the Grand Challenge

The 2005 Grand Challenge was a 212 km race through the Mojave Desert. To win the challenge, a team's robot had to complete the course in less time than any other robot, and do so within 10 h. Information about the route and exact distance of the challenge was withheld until 2 h prior to race start, precluding prerunning or recording of the race route.

Figure 1. Sandstorm and H1ghlander were developed to navigate at high-speed in desert terrain.

The route description contained a series of waypoints marking the corridor and speed limits within which the robots were required to travel. Once away from the starting line, the robots were required to be completely autonomous. The only communications allowed to a robot were publicly available global positioning system (GPS) signals and a safety kill system used by challenge personnel to ensure safety.

The Red Team developed two robots and used a combination of autonomous and human preplanning to complete the Grand Challenge. Once underway, the robots used onboard sensors to adjust a preplanned route, to avoid obstacles and to correct for errors in position estimation.

The approach presented in this article proved successful; both Sandstorm and H1ghlander completed the Grand Challenge and demonstrated a new level of robust high-speed navigation.

1.2. Overview

This article describes the Red Teams vehicles, software, and testing process in depth. Section 2 begins the discussion with a detailed description of the electromechanics that make up both robots, and explains the significant differences between them. Section 3 describes the software architecture and components that operate onboard the robots including algorithms for perception, navigation and tracking. Both the benefits and limitations of the utilized approaches are described. Section 4 provides a short description of the preplanning system, while Section 5 provides a detailed discussion of the various testing processes employed by the team during the development cycle. In Section 6, an analysis of how the Red Team robots performed during the Grand Challenge event is presented. Section 7 closes the discussion with a number of lessons learned and some ideas for future work building from the Grand Challenge.

2. THE ROBOTS

Sandstorm and H1ghlander are substantially different robots; each has its own unique actuation and control challenges. While much hardware and software is shared between the platforms, each vehicle has distinct performance characteristics.

The decision to develop two robots was not made lightly. The extra time, effort, and personnel necessary to build and maintain two similar, but not identical robots represented a significant concern. However, operating two robots provides two principal advantages. During testing, the potential availability of two robots helps ensure that there is at least one operational for software testing. Increased testing time is one of the keys to increasing robustness and capability. During the challenge, two robots provide an increased level of reliability since even if one of them fails, the other will continue. In the team's estimation, the advantages of having two robots for testing and racing outweighed the concerns.

Figure 2. Sandstorm before and after modifications.

2.1. Chassis

Sandstorm is a modified 1986 Model 998 HMMWV. The vehicle has been highly customized for autonomous control. The roof and passenger compartments have been removed in favor of a large electronics enclosure. Pictures of Sandstorm before and after the vehicle modifications are included in Figure 2. The

electronics box is suspended atop the vehicle platform on 12 coil over damper struts. Suspension lowers the natural frequency of the electronics enclosure so that sensors can be rigidly mounted. It also minimizes the shock dose to the computers and electronics so they are protected from severe accelerations. Sandstorm's control frame floats with the electronic enclosure, causing the navigation system to drive the floating electronics box, without a full understand of the position and orientation of the chassis. This is one of the main causes of Sandstorm's characteristic smooth, but slightly sloppy driving style.

H1ghlander is built from a 1999 commercial H1 truck chassis with an upgraded 2001 electrical system. This upgrade includes a new vehicle harness, engine controller, and transmission controller. These were installed to allow easy reprogramming of engine and transmission functions. H1ghlander also uses an upgraded hydraulic steering system which provides a quick high accuracy steering response. All other vehicle components remain stock with the exception of a race quality suspension similar to Sandstorm's. Unlike Sandstorm, H1ghlander retains three of its four passenger seats. This allows team members to ride in the vehicle for development, and also gives other people the unique experience of riding in an autonomous vehicle. Figure 3 shows H1ghlander before and after its vehicle modifications. Because H1ghlander does not have a floating electronics enclosure, the navigation system has a better estimate of the true pose of the vehicle. This helps H1ghlander drive more crisply than Sandstorm.

Figure 3. H1ghlander before and after modifications.

2.2. Controls Strategy

Throughout the vehicle system, feedback controllers are used in order to regulate systems and position actuators. In most cases, the method used is a variant of the proportional integral derivative (PID) controller. While PID control is not always the most accurate or highest performance controller, it is easy to use and robust. Equation (1) represents the general PID controller:

$$u = K_p e + K_i \int e dt + K_d \frac{d(-PV)}{dt}. \quad (1)$$

In general, there are no accurate models available (or the time needed to develop them) for most of the vehicle systems, which makes more complicated control difficult. All PID algorithms are implemented in discrete time through the use of real-time processes running with fixed time steps. The use of a simple and easy to tune control strategy across the entire vehicle helps ensure the reliability and robustness of these systems.

The vehicle electronics systems for both Sandstorm and H1ghlander utilize multiple electronic control modules (ECMs). ECMs are automotive or commercial grade components that contain one or more embedded processors, including the input/output circuitry, and memory necessary to carry out a given task or function. ECMs are distributed throughout the vehicle system, and are interconnected by a series of automotive grade data links.

Each ECM runs a real-time operating system in order to ensure that the different control and communications loops occur at deterministic rates, and

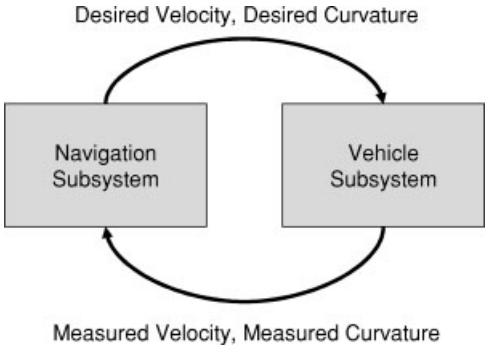

Figure 4. Vehicle to navigation system interface.

that safety critical tasks are performed reliably. Software routines, such as boot code, communications, and input/output functions were developed, using a combination of hand coded C or assembly language. Control routines were developed using Simulink models, which were then autocoded into functions that were run on the control modules. This allowed rapid development and testing of controllers in simulation, before autocoding them to run on the vehicles.

There is a clean break in functionality between vehicle system and navigation software, which allowed the use of a very simple interface between the two subsystems. This interface, as shown in Figure 4, allows a navigation process to command a vehicle velocity and curvature. The vehicle system reacts to those commands and provides measured velocity and steering position back to the navigation system. The vehicle system also safeguards the vehicle via emergency-stop radios and deactivation buttons.

2.3. Power and Cooling

One of the significant differences between Sandstorm and H1ghlander is their power generation and cooling systems. Neither vehicle has the factory alternator installed, but both require about 4 kW of auxiliary power for all of the necessary computing, sensing, and actuation components. Sandstorm employs an auxiliary generator mounted on the aft of the vehicle. The 24 V generator is turned by a small diesel engine which shares the propulsion engine's fuel supply. A compressor mounted to the propulsion engine provides cooling for an evaporator mounted in the electronics enclosure.

H1ghlander's power system is more complex. It incorporates a switch-reluctance generator driven by the main propulsion engine that generates between 4 and 7 kW (depending on engine rpm) at 340 Vdc. This high-voltage bus is used to efficiently power H1ghlander's electric air conditioning compressor, and is downconverted to both 12 V and 24 V in order to supply the necessary vehicle and electronics power.

Both vehicles' power systems are controlled by ECMs which contain embedded processors and input and output circuitry to monitor and control their respective power components. The power control algorithms are very similar. They use a modulated voltage to control current technique that was developed in order to maintain acceptable power levels even at times when current draw exceeds available power. This control scheme, as outlined in Figure 5, uses predominantly integral (I) or "follower" control with offset gains and feedback derived from logic based on measured current and voltage. The controller is designed to operate with batteries in parallel with the output voltage buses. A dual/switching controller is needed because at steady state the generator is able to provide more than enough power to supply the components, but when the batteries are at a low state of charge, the current draw required to charge the batteries can exceed the available power, causing an overcurrent or "stall" condition at the generator. To compensate for this condition, the power controller is designed to change the output voltage of the generator in order to keep the current draw to acceptable levels. By lowering the voltage, the difference between the generator and batteries can be decreased. While in this "max-current" case, the output voltage is modulated to keep current draw at about 90% of available current. As the batteries charge, the required current drops and the controller switches to voltage control and maintains an ideal voltage level. This basic power control method has proven to be robust, and has been extended in H1ghlander to control the power consumption of two downconverters and the cooling unit.

For both Sandstorm and H1ghlander, the power systems are designed to seamlessly function through minor power generation glitches. For example, during testing Sandstorm has traveled over 125 km without any power production from the auxiliary generator, demonstrating the robustness of this design.

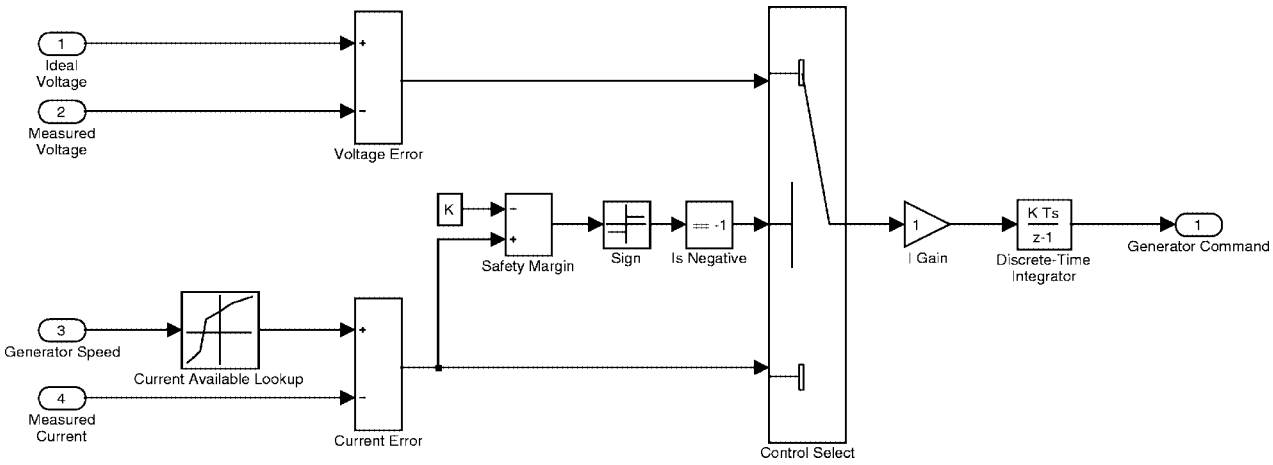

Figure 5. H1ghlander's power controller.

2.4. Steering

Electronic actuation of steering is a fundamental part of autonomous vehicle control. Sandstorm and H1ghlander have vastly different steering actuation strategies, and in turn have very different steering performance characteristics. In both cases, the systems respond to steering curvature commands from a tracker in the navigation software. The commanded curvature is linearly mapped to a steering angle in the controller, which is then maintained. There is no feedback control around actual curvature, only around steering angle. This proved a challenge to the vehicle's tracking system since it does not account for wheel slip, caused by differing ground conditions, or mechanical sensor slip which inherently changes the mapping between curvature and steering angle.

Mechanically, Sandstorm retains its complete stock steering system. To electronically steer the wheels, a large driven gear is mounted to the top of the steering column, behind the steering wheel. A drive gear, attached to a dc motor and harmonic drive gear set [shown in Figure 6(a)], is mated with the steering column gear. The harmonic drive gearing provides a very high gear ratio with zero backlash and large amounts of torque. The downside of this high-reduction gearing is that it limits steering speed.

The motor is controlled through a drive amplifier by an ECM, which runs a closed-loop control algorithm around the steering angle. Controller feedback is provided by a rotational sensor mounted to the output shaft of the power-steering gearbox, which outputs a pulse-width modulated signal proportional to steering position. For robustness, there is also a multiturn sensor that measures position at the motor. A PID controller is used to maintain wheel steering position by outputting motor torque

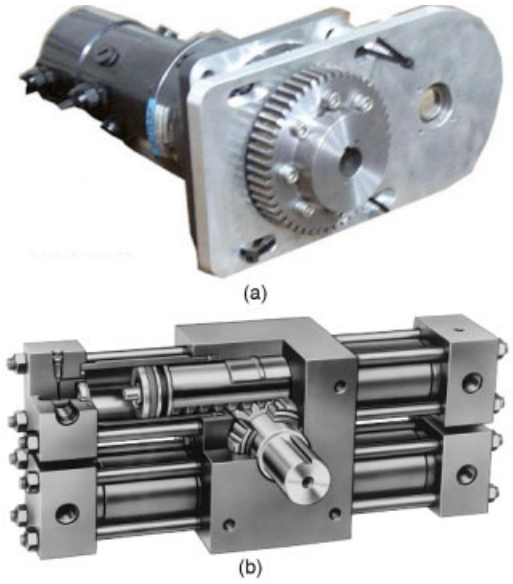

Figure 6. (a) Sandstorm's steering actuator and (b) H1ghlander's hydraulic steering actuator.

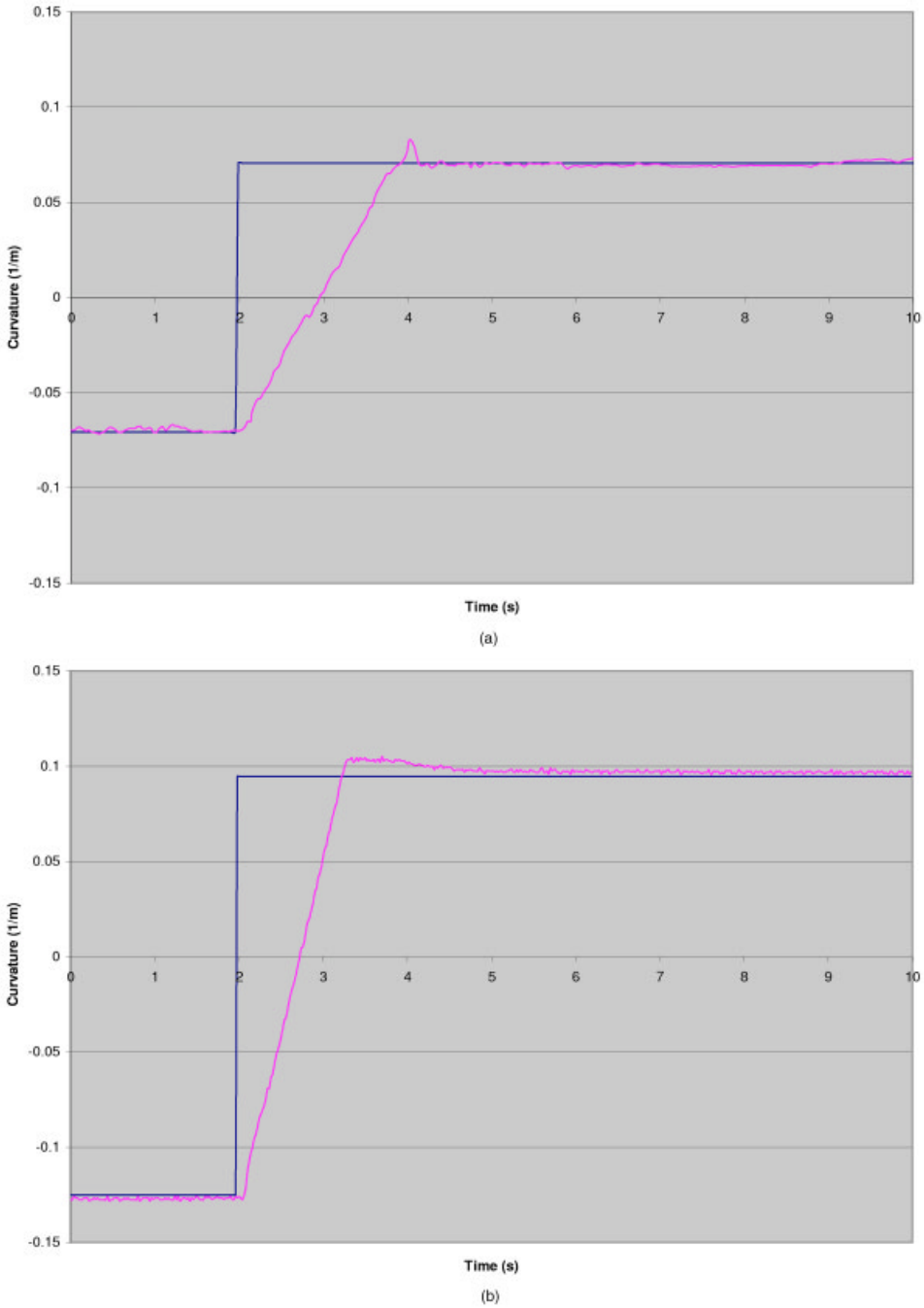

Figure 7. Plots of Sandstorm (a) and H1ghlander (b) steering response.

and reading the steering angle. This steering approach retains a majority of the stock steering system, which makes the system simple and robust. The downsides include the limited steering actuation speeds and limited accuracy due to a large mechanical deadband in the power steering linkage, which causes hysteresis in the controller.

H1ghlander employs a different steering strategy; all of the stock steering components were removed and replaced with a full hydraulic steering

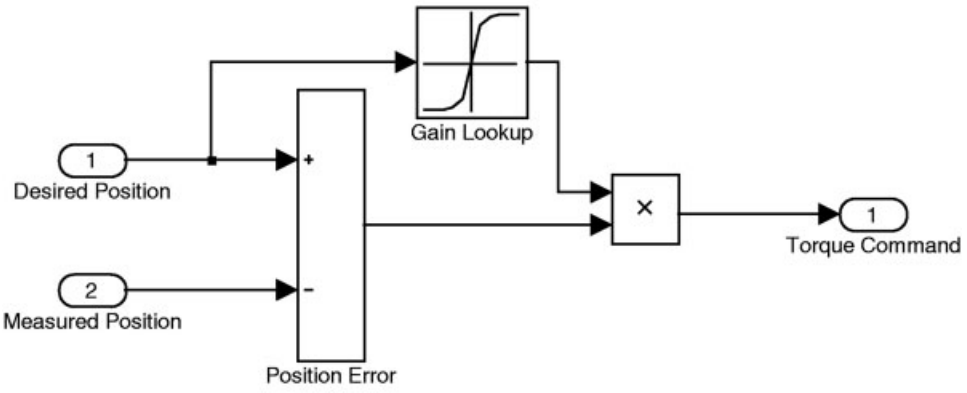

Figure 8. Model of the brake controller.

system. The hydraulic system is composed of a dual-cylinder rotary hydraulic actuator [shown in Figure 6(b)], a fixed-displacement hydraulic pump, and an electrohydraulic valve to control the hydraulic flow.

Electronics in the valve maintain a closed-loop control of the valve's spool position. Spool position is directly proportional to hydraulic flow (which can be mapped to cylinder velocity) and is commanded by an ECM. Steering angle is measured in the rotary actuator both by measuring the rotary output shaft position, and the linear position of one of the hydraulic cylinders. The ECM reads these positions, selects which one to use for feedback, and outputs a desired spool position based on a PID control algorithm. The advantage of this steering strategy is very responsive steering, and the ability to hold a very precise steering angle. The downside is the complexity of a hydraulic system, which is prone to leaks, heat, and filtration issues, each of which was encountered during the development.

Figure 7 illustrates the differences in steering response between Sandstorm and H1ghlander.

2.5. Velocity Control

The control of vehicle velocity is an important aspect of high performance driving. In a racing atmosphere, speed control must be accurate and responsive as it is constantly being adjusted to ensure vehicle stability. Velocity also poses a controls challenge, since it involves two different mechanical systems (propulsion engine and brakes) to maintain speed in any number of environmental conditions.

Sandstorm has a mechanically controlled engine.

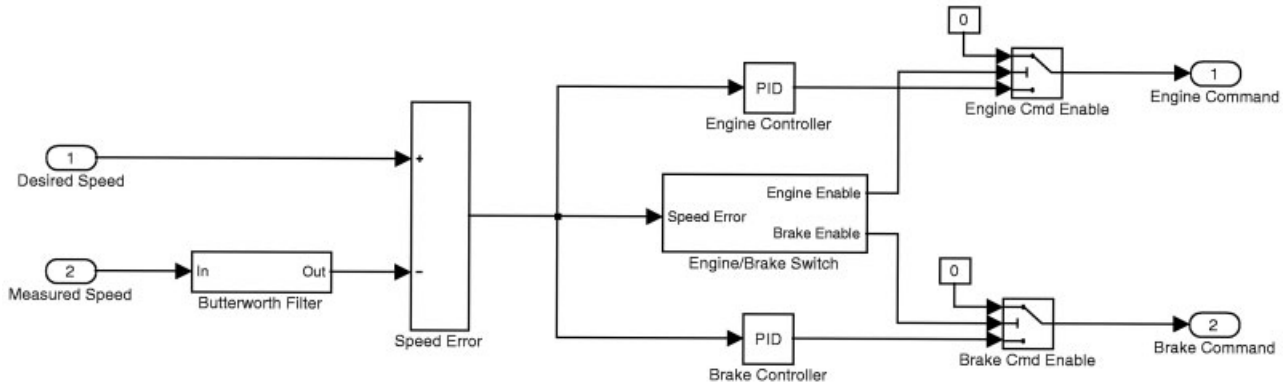

Figure 9. Model of the speed controller.

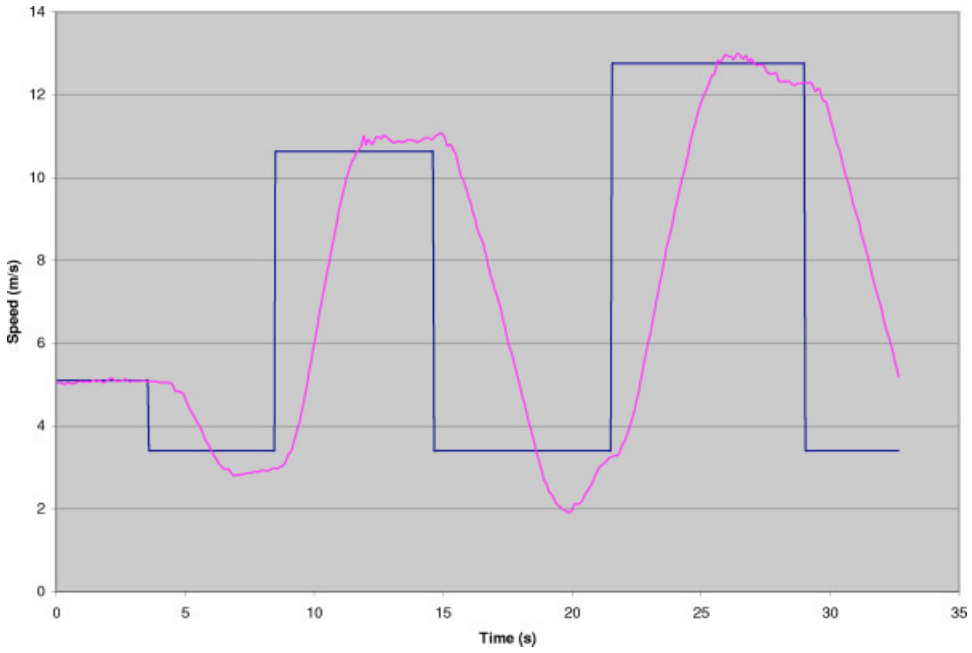

Figure 10. A plot of speed controller performance.

This means that to actuate the throttle, a valve on the injection pump must be physically turned. To accomplish this, an automotive-grade throttle body actuator was modified and mounted to the injection pump. The actuator is a simple dc motor with analog position feedback. An ECM reads this position and runs a PID closed-loop control algorithm in order to command the injection pump to a specific throttle level.

In contrast, H1ghlander's engine is fully electronically controlled, meaning that its entire operation, from fuel injection to timing is commanded by an electronic engine controller. This makes autonomous activation very simple; a message is sent across a data-link and acted on by the engine controller.

Both Sandstorm and H1ghlander use the stock service brakes to slow the vehicle. In both cases the service brakes are actuated by an electric motor. Both motors are three phase brushless design with an integral 50:1 harmonic drive gear reduction. In Sandstorm's case, the motor is mounted to press on the brake pedal. This results in a relatively slow braking response but provides significant mechanical advantage. In H1ghlander, the motor is mounted to actuate the brake master cylinder directly. This mounting achieves quicker response, since less motor travel accounts for more braking force. In both configurations an ECM runs a proportional controller to command braking, which effectively provides torque-based control of the motor. This type of control inherently compensates for system degradation, such as brake wear or different pressure line losses. A diagram of Sandstorm and H1ghlander's brake controller is given in Figure 8.

The speed controller is a piece of embedded software that receives desired speed from the navigation system and commands the throttle and brakes to maintain that speed. The speed controller is actually comprised of three controllers: A speed controller using a throttle command, a speed controller using a brake command, and transition logic to determine which one to use. The control strategy mimics a layman human driver, where only the engine or the brakes will be actuated at any time. This is contrary to the driving strategy of most racers, who often times actuate the throttle and brakes simultaneously. Figure 9 illustrates this speed controller graphically. The throttle and brake controllers each use a proportional integral control scheme. Transition logic integrates the speed error and uses that as a time-delay switch with limits depending on commanded speed and whether the system is switching from the throttle to the brakes or vice versa.

Table I. A qualitative comparison of sensors considered for inclusion in the navigation system.

Sensor	Advantages	Disadvantages	Selected
Conventional camera	- Wide vertical field of view - Large swath of dense data	- Data quality decreases due to lighting changes and glare - Conventional processing techniques are challenged offroad	No
Stereo camera	- Wide vertical field of view - Generates dense three-dimensional data sets	- Data quality decrease due to lighting changes and glare - Measurement accuracy decreases as the square of range	No
LIDAR	-Accurate range measurements - Straightforward to integrate - Wide horizontal field of view	- Provides only a single plane of data -Correlating between line scans may be challenging when moving over rough terrain	Yes
Automotive RADAR	- Commercial grade hardware - Integrated signal processing to identify targets - Operates through most visual obscurants	- Off-road performance is not characterized - Output provides only limited information for external processing	No
Navigation RADAR	- Dense raw signal strength output - Operates through most visual obscurants	- Understanding raw RADAR signal returns is complicated	Yes

This method allows more error at higher commanded speeds to minimize changeover between throttle and brakes, but also generates a quick response for sharp deceleration before turns. The speed feedback used in the control algorithms is obtained from the transmission and smoothed using a third-order Butterworth filter.

The speed controller maintains speed to within 0.5 m/s on average, with a 1–2 m/s undershoot when braking (response shown in Figure 10). This undershoot is due to a nonconstant contact point in the brake system which causes the braking force to increase dramatically. The controller does not compensate for this nonlinear braking response since the error was determined to be acceptable.

2.6. Perception Sensors

Sandstorm and H1hghlander combine data from a variety of sensors to perceive the world. In selecting sensors, the team evaluated monocular cameras, stereo cameras, LIDAR, and RADAR systems to find modalities amenable to generating terrain evaluations under the difficult conditions of the Grand Challenge. Table I outlines the sensing modalities considered and the advantages and disadvantages for this problem.

These considerations led to a perception strategy based on a set of five LIDAR and a navigation RADAR. Three of the LIDAR operate to characterize terrain, using overlapping field of view to provide redundancy. The two remaining LIDAR and the RADAR are used to detect obvious obstacles at long ranges. Figure 11 illustrates the sensor fields of views, while Figure 12 shows the sensor locations on the robots. This design provides a robust perception suite, with multiple sensors observing the significant portions of terrain in front of the robots. The remainder of this section describes the specifics of the sensors selected for the robots. Table II presents the specifications of the sensors used on the robot.

2.6.1. LIDAR

A Riegl Q140i scanning laser range finder is used as the primary terrain perception sensor for both robots due to its long sensing range, ease of integration, and few, well understood, failures modes. A limita-

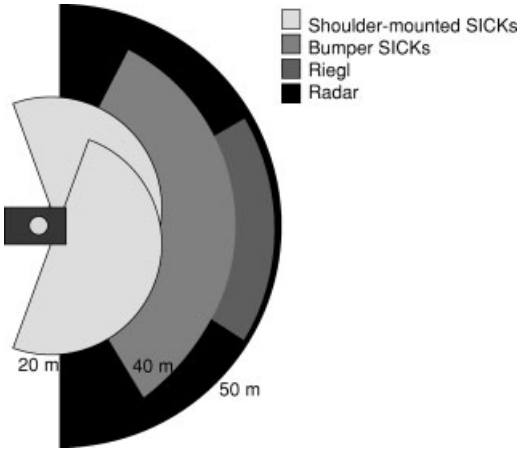

Figure 11. Sandstorm and H1ghlander have multiple sensors with overlapping fields of view.

tion of scanned LIDAR is that it is generally only possible to collect dense point data in a single plane. Flash LIDAR does not suffer this limitation but is still too range limited to be useful at Grand Challenge speeds. Two-axis mechanically scanned LIDAR have reasonable range, but cannot scan rapidly in both axes and thus do not provide significant benefits over a single scanning plane for this application.

In addition to the long-range LIDAR, four SICK LMS 291 laser scanners are used to provide short-range supplemental sensing. Two are mounted in the front bumper, providing low horizontal scans over a 120° wedge centered in front of the robot.

Table II. Characteristics of navigation sensors.

Sensor	Characteristic
Narrow field of view LIDAR	Horizontal field of view: 60° Instantaneous field of view: 0.17° × 0.17° Dots per scan: ~240 Scan Rate: 50 Hz Manufacturer: Riegl Model: Q140i
Wide field of view LIDAR	Horizontal field of view: 180° Instantaneous field of view: 0.86° × 0.86° Dots per scan: 181 Scan Rate: 75 Hz Manufacturer: SICK Model: LMS 291
Navigation RADAR	Horizontal field of view: 360° Instantaneous field of view: 1.2° × 4.0° Measurements per scan: 300 Scan Rate: 2.5 Hz Manufacturer: NavTech Model: DS2000

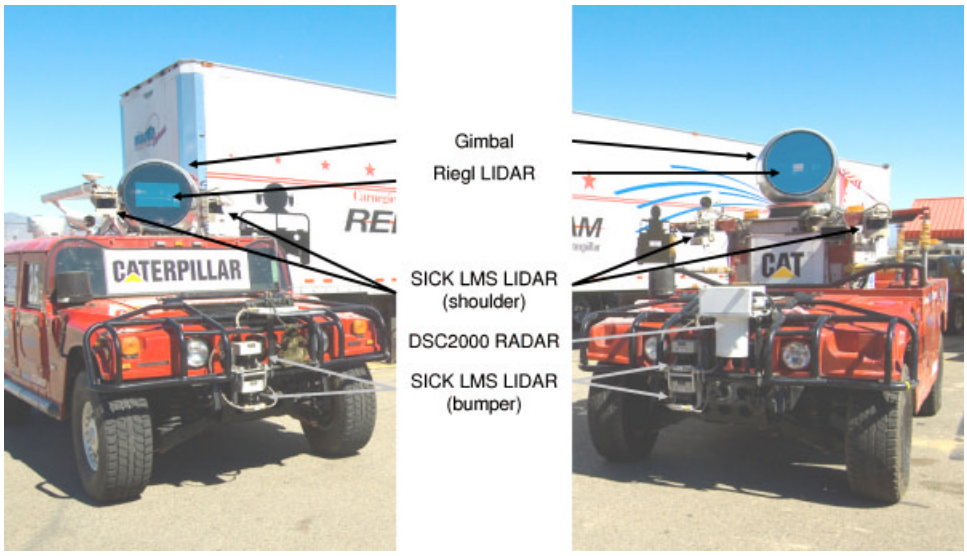

Figure 12. H1ghlander (left) and Sandstorm with their sensors labeled.

Figure 13. Front and back view of the gimbal.

These sensors can be used to detect obvious, large, positive obstacles. The other two SICK LMS laser scanners are mounted to the left and right of the vehicle body. These sensors perform terrain classification.

2.6.2. RADAR

While LIDAR may have difficulties sensing in dusty environments, RADAR operates at a wavelength that penetrates dust and other visual obscurants but provides data that are more difficult to interpret. Because of its ability to sense through dust the NavTech DS2000 continuous-wave frequency modulated (CWFM) radar was used as a complimentary sensor to the LIDAR devices.

2.6.3. Pose Estimation

Reliable and robust position sensing is essential since it is central to performing reliable control and building usable world models. The implementation of position sensing is a major undertaking that can drain valuable development resources. To avoid this problem, Sandstorm and H1ghlander use an off-the-shelf pose estimation system. The Applanix M-POS provides position estimates by fusing inertial and differential GPS position estimates through a Kalman filter. The output estimate is specified to have submeter accuracies, even during extended periods of GPS dropout. The M-POS system also provides high accuracy angular information, through carrier differencing of the signal received by a pair of GPS antennas, and the inertial sensors. The M-POS system outputs a pose estimate over a high-speed serial link at a rate of 100 Hz. This constant stream of low-latency pose information simplifies the task of integrating the various terrain sensor data sources.

2.6.4. Stabilization and Pointing

The ability to interpret data from long-range sensors, such as a Riegl LIDAR scanner, can be severely hampered by pitching and rolling induced by robot motion over terrain. The performance of the single axis scanning LIDAR is particularly affected by mechanical excitation in the pitch axis. When sensing at reasonably long ranges, even small-scale pointing errors can result in a dramatic change in where a sensor's beam intersects the terrain. Because of this, range data associated with small obstacles or terrain details at distance become ambiguous, and the overall perception performance is severely degraded. Active attenuation of the terrain excitations can reduce this effect, yielding interpretable terrain range data.

The Gimbal sensor mounting design aligns the Riegl LIDAR's optical aperture with the center of the gimbal and balances the mass distribution around the rotational center. Each axis includes minimal-mass components and the simplest possible gimbal

Table III. Gimbal characteristics.

Parameter	Value
Payload compliment	High-resolution LIDAR line scanner
Payload dimensions	240 mm × 250 mm × 500 mm
Payload weight	12+ kg
Platform weight	~25 kg
Peak power	550 W

support structure with the design goal of minimizing the moment of inertia. By minimizing the moment of inertia, the overall responsiveness of the gimbal is increased.

Harmonic drive actuators are used to point the gimbal based on feedback from a combination of incremental and absolute position encoders, and fiber-optic gyros. Each gimbal axis assembly is designed for electrical and mechanical simplicity. To achieve this goal, common design and components are used on each of the axes. The aluminum bracket components are designed to have minimal mass and moment of inertia about their respective rotational axes, while still maintaining sufficient strength and stiffness.

The entire gimbal mechanism is enclosed within a protective carbon fiber shell. The shell prevents water and dust from damaging the mechanism and electronics, and includes a front window with specially coated optical glass for the sensors to operate through. The fully assembled gimbal is shown in Figure 13. Table III describes the gimbal characteristics, while Table IV describes its performance.

3. ONBOARD NAVIGATION SOFTWARE

Onboard navigation software combines incoming sensor data with a preplanned route to generate a new safe and traversable route. In the following sections, the architecture and algorithms used to drive Sandstorm and H1ghlander are presented.

Table IV. Gimbal performance characteristics.

Axis	Range of motion (°)	Angular velocity (°/s)	Acceleration (°/s^2)
Pitch	±40	360	49,500
Roll	±40	360	4,200
Yaw	±90	360	1,450

3.1. Architecture

The navigation software-architecture was designed with the infrastructure to support high-speed navigation while being robust to sensor failures and adaptable enough to support a rapid, relatively unstructured development process. These design goals led to a pathcentric navigation architecture, built around a set of well-defined rigid data interfaces.

In this pathcentric architecture (see Figure 14), the fundamental command action is to execute a path. This differs from a majority of autonomous navigation architectures which use an arc as the fundamental action. The path data structure is pervasive throughout this approach; the preplan is provided as a path, path-planning acts as a filter on the path, and the perception system uses it to steer sensor focus and account for incompletely sensed terrain.

The pathcentric architecture has several advantages that improve performance and robustness over arc-centric architectures (Simmons et al., 1995; Kelly & Stentz, 1997; Betulla, Manduchi, Matthies, Owens & Rankin, 2000; Biesiadecki, Maimone & Morrison, 2001; Urmson, Dias & Simmons, 2002). It provides a simple method for incorporating human input through a preplanned route. Given a route, planning can be performed in a fixed width corridor around the preplanned route, thus reducing the search space for a planning algorithm from the square of the path length to linear in the path length. The pathcentric approach avoids problems with arc-based arbitration, such as discontinuities in steering commands (due to contradictory information) and jerky control (due to discrete arc sets). Furthermore, since the navigation system commands the execution of paths rather than discrete arcs. This effectively decouples the steering control frequency from the planning frequency, further increasing the smoothness of control. This makes the system insensitive to reasonable amounts of variation in the planning cycle, further increasing robustness.

To use terrain evaluation data from multiple sources, the architecture uses a map-based data fusion approach. To provide this functionality, the architecture defines a second fundamental data type; the map. In this system, a map is a rectilinear grid

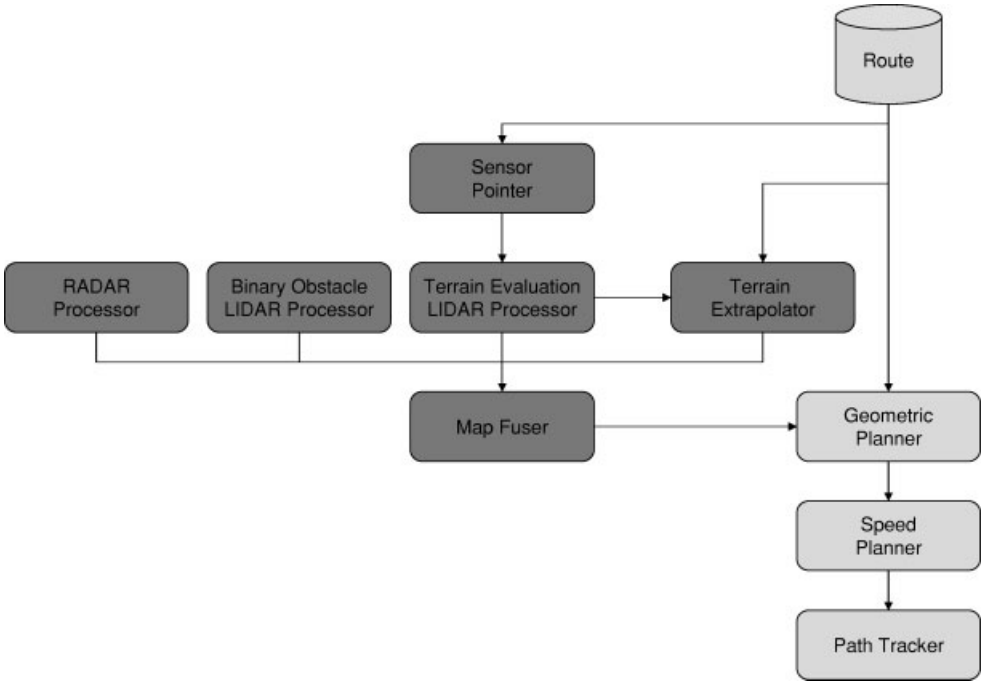

Figure 14. The pathcentric architecture of the onboard navigation software.

aligned with the world coordinate system and centered on the robot. Each of the sensor processing algorithms produces its output in the form of a cost map. Cost maps are a specific map type that represents the traversability of a cell using a numeric value. In this implementation, the cells have an edge length of 25 cm. Figure 15 shows an example cost map.

The path and cost map are two of a handful of fundamental data types (other examples include vehicle pose and LIDAR line scan data structures) that are used as the syntax for communication between various data processing modules. The software implementation uses a communication and infrastructural toolset that allows algorithm developers to create modules that communicate with the rest of the system using the specified data types through a set of abstract, reconfigurable interfaces (Gowdy, 1996). During development and debugging, the interfaces for an algorithm can be configured to read data from time-tagged files using a common set of data access tools. As an algorithm matures, the interfaces are reconfigured to communicate with the rest of the navigation system. This approach helped reduce the required uptime and availability of the robot.

By using a common set of carefully defined and strictly controlled data types as the syntax for communication, it is possible to quickly develop new features for either path or map processing. While the syntax is defined and controlled, the semantics, or meaning, of the data being passed between modules is free to be adapted as new ideas evolve and algorithms are developed. This flexibility makes the overall system robust and adaptable to ever-evolving ideas that develop as the navigation problem is explored.

3.2. Sensor Fusion

The map fusion process is critical to the robustness of the navigation system, as it enables the system to cope with sensor failures and missing data. To use the data from the various sensor processing algorithms, it is necessary to combine it into a composite world model (either implicitly or explicitly). In this system, the data are combined in the sensor fusion module by generating a composite map using a weighted average of each of the input maps.

Each of the processing algorithms specifies a confidence for the output map it generates. The fusion algorithm then combines the maps with these

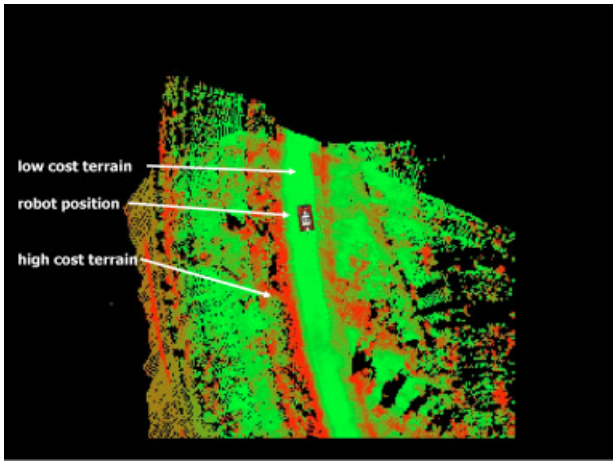

Figure 15. An example cost map showing low (light) and high (dark) cost terrain.

weightings to generate the composite expected cost map. This design allows the sensor processing algorithms to adjust their contribution to the composite map if they recognize that they are performing poorly. In practice, a set of static weights, based on a heuristic sense of confidence in the algorithms ability to accurately assess the safety of terrain, worked well. With calibrated sensors, this approach produces useable composite terrain models. Figure 16 shows various input maps and the resulting, fused composite map.

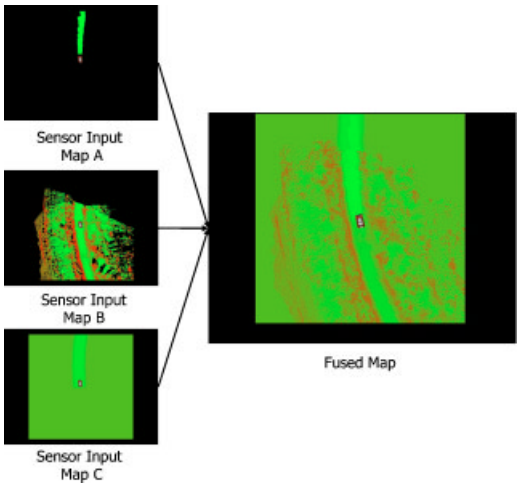

Figure 16. An illustration of fused sensor maps.

3.3. Perception

In this approach to high-speed navigation, three principal risks are considered: Hitting large obvious obstacles that can destroy a vehicle, driving on avoidable rough terrain that will damage a vehicle over prolonged periods of time, and dynamic effects—such as sliding and rollovers—which cause a loss of control and can also potentially destroy a vehicle. The perception algorithms presented here are designed to address these risks. The binary obstacle LIDAR and RADAR processors are designed to quickly detect obvious obstacles at range. The terrain evaluation LIDAR processor is designed to generate a continuous valued classification of terrain, ranging from safe and smooth to intraversable. The slope calculations in this algorithm are used to steer the robot away from terrain with a likelihood of causing static tipover, but falls short of estimating dynamic tipover. Instead, the risk from dynamic effects is mitigated in a speed planning algorithm.

The various perception algorithms provide an overlapping (both geometrically and in terms of capability) set of models that reduce the likelihood of missing the detection of any obstacles while also providing robustness in the face of a sensor or algorithmic failure.

3.3.1. Sensor Pointing

The sensor pointing algorithm uses a preplanned path as a guide as to where to point the Riegl LIDAR. A priori knowledge of the path enables the algorithm to point the sensor around corners, prior to the robot making a turn, and helps the perception system build detailed models of terrain in situations where the fixed sensors would generate limited information. A simple algorithm calculates a look-ahead point along the path given the current pose and speed of the robot. The look ahead point is then used to calculate the pitch, roll, and yaw required to point at this location. These commands are then passed onto the gimbal. The data generated by the pointed and fixed shoulder mounted LIDARs are used by the terrain evaluation LIDAR processing algorithm.

3.3.2. Terrain Evaluation LIDAR Processing

Terrain classification and obstacle detection are at the core of high-speed outdoor navigation. The ter-

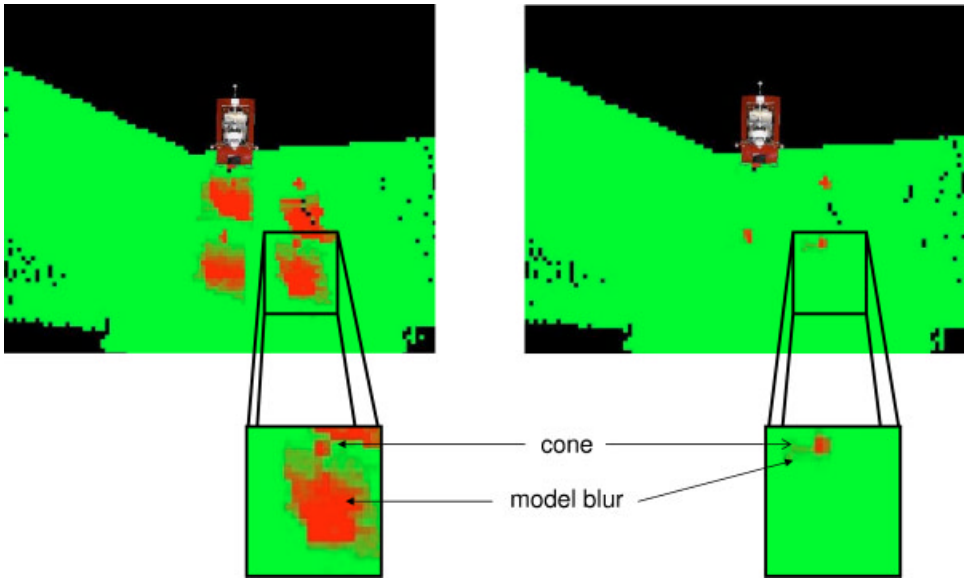

Figure 17. Obstacle blur before (left) and after foreground separation filter is applied.

rain evaluation system borrows ideas from Kelly and others (Kelly & Stentz, 1997,1998; Batavia & Singh, 2002; Kelly et al., 2004) in performing terrain evaluations within a single line scan to reduce the effects of imperfect pose estimation.

The terrain evaluation approach is derived from the Morphin algorithm (Simmons et al., 1995; Golberg, Maimone & Matthies, 2002; Urmson et al., 2002) but has been adapted to operate on a single line scan of data instead of a complete cloud. The algorithm operates by fitting a line to the vertical planar projection of points in vehicle width segments. The slope and chi-squared error over this neighborhood of points provide the basis for evaluation. The operation is performed at each LIDAR point in a scan. If a point does not have a minimum number of points within a support distance or the surrounding points are not sufficiently dispersed, the point is not classified. The traversability cost is calculated as a weighted maximum of the slope and line fit residual.

Once traversability costs have been determined, each point is projected into a cost map, with independent cost maps maintained for each sensor. The terrain evaluation from each sensor is periodically combined into a composite output map. The traversability cost for each cell in the composite map is computed as the weighted average of the costs from each sensor, with weights proportional to the number of points used in generating the traversability cost for each sensor map.

While this basic algorithm works well, it blurs small obstacles over a large area since it does not separate foreground obstacles from background terrain (see Figure 17). The foreground obstacle pixels cause the line fit for the background pixels to have significant residual errors. To address this problem, a filter is used to separate foreground features from background terrain. During the evaluation process, any point at a significantly shorter range than the point being evaluated is ignored. This has the effect of removing discrete foreground obstacles (and their contribution to the line fit residual) from the evaluation of background terrain, while still correctly detecting obstacles. Figure 17 illustrates the effect of this filtering on a scene consisting of four cones in a diamond configuration. Without filtering, each of the four cones is represented as obstacles the size of a car, with the filtering applied the cones are represented as obstacles of the approximately the correct size.

3.3.2.1. Limitations

There are two situations where the terrain evaluation produces incorrect output: When LIDAR scans

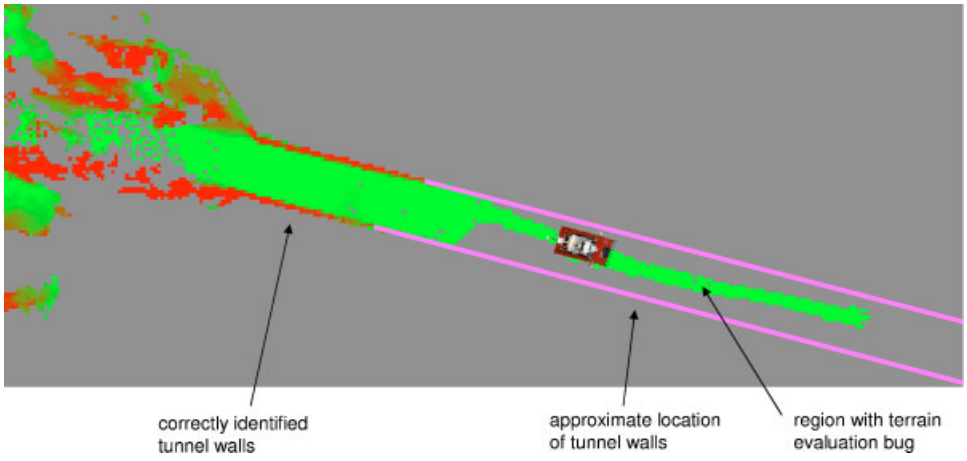

Figure 18. An example of where a constrained environment causes the LIDAR terrain evaluation algorithm to behave poorly.

graze a surface, and when traveling through narrow tunnels or canyons.

In grazing scenarios, the algorithm will misclassify grazing returns as obstacle locations. The grazing returns are classified as obstacles because the shape of the LIDAR return is indistinguishable from a return with significant obstacles. Fortunately, this occurs rarely and is mitigated through the use of multiple sensors. Applying a nearest-neighbor clustering approach (Batavia & Singh, 2002) may be a solution to this problem.

The second problem of operation in very constrained environments (such as narrow tunnels) results in areas with sensor data not being classified. This problem stems from the requirement that the terrain evaluation only occur at LIDAR points that have neighbors with a minimum dispersion. This requirement causes a one-half vehicle width of data at each end of the LIDAR scan to be ignored. In normal outdoor navigation scenarios, this defect is insignificant. In situations where the LIDAR returns all occur within a narrow area (e.g., a tunnel), important terrain features are not classified since the points that make up the walls of the tunnel do not have sufficiently dispersed neighbors. This problem is illustrated in Figure 18. Note that the approach to the tunnel and tunnel walls near the opening are classified correctly, but as soon as the sensor measurements become constrained within the tunnel, only the center of the tunnel is classified while the outer one-half vehicle width on either edge of the scan are not classified, including the walls.

Both of these shortcomings are mitigated by the combination of the other terrain evaluation algorithms and the terrain extrapolation modules described in the following sections.

3.3.3. Binary Obstacle LIDAR Processor

The binary obstacle LIDAR Processor is designed to quickly and robustly detect obstacles by collecting points over short time periods while a vehicle drives over terrain. The algorithm uses the fact that LIDAR points cluster on vertical faces to detect obstacles.

Using geometric information to determine a binary measure of traversability is common and has been a topic of research for decades. Typically, this information is gleaned from images using classification techniques (Ulrich & Nourbakhsh, 2000) or geometric analysis of three-dimensional (3D) terrain data (Singh & Keller, 1991; Talukder, Manduchi, Rankin & Matthies, 2002). In recent work (Roth, Hamner, Singh & Hwangbo, 2005) have extended the RANGER algorithm to use LIDAR point clouds accumulated as a vehicle drives. The algorithm derives roughness, roll, and pitch by fitting planes to patches of point clouds.

As a LIDAR is moved through space, it sweeps the terrain, and a point cloud representing this terrain is built by registering each scan using the ve-

hicle and sensor pose. Selected pairs of points from this cloud are compared to compute the slope and relative height of the terrain.

Traversability is determined by performing a point-wise comparison of points within a region surrounding the point in question. If the slope and vertical distance between the two points is determined to be greater than a threshold value, both points are classified as obstacles. Given two points to compare, slope is computed as

$$\theta = \tan^{-1}(|\Delta z|, \sqrt{\Delta x^2 + \Delta y^2}). \quad (2)$$

If Δz and θ are greater than threshold values, an obstacle is inserted into the cost map.

Comparison of full rate LIDAR data is computationally expensive (Lalonde, Vandapel & Hebert, 2005). To make comparison rates reasonable, points are binned into two-dimensional (2D) (x,y) cells and hashed by 2D cell location. Each hash location contains a list of all points within a cell. When a new point hashes to a hash location containing a list of points far from the new point, the list of points is cleared and the new point is inserted. The data structure allows near-constant time comparison of nearby points by doing a hash lookup in the region of a point of interest.

With long-range sensors, small errors in attitude of the sensor cause large errors in point registration. The comparison of two measurements of the same terrain patch from two different view points can falsely generate an obstacle if the vehicle pose estimate is erroneously pitched or elevated. Furthermore, the likelihood of inconsistent pose estimates is increased with time and distance traveled. Thus, it is important to delete old points. To accommodate fast deletion, points are inserted into a ring buffer in the order that they are received. Once the ring buffer is full, each new point overwrites the current oldest point in the buffer and the hash table.

3.3.3.1. Limitations

This algorithm relies on excellent registration of sensors to the world. The calibration of sensors relative to the vehicle center is a significant cause of error when multiple sensors are compared. Relative errors in pose estimation can cause false positives between scans of a single sensor. These two considerations limit the effectiveness of this algorithm at range.

Because of the reliance on registration, sensors that are not rigidly mounted relative to the vehicle coordinate frame cannot be processed using this algorithm. On H1ghlander, a considerable amount of capability is gained by comparing the bumper mounted SICKs. In flat terrain, these sensors can detect large obstacles (fence posts, large rocks, cliff walls, etc.) at up to 50 m. Because of its floating electronics box, Sandstorm cannot use the bumper mounted sensors with this algorithm.

Since this algorithm is designed to complement the LIDAR terrain evaluation algorithm, it is not sensitive to terrain features, such as roughness and gentle slopes. The algorithm misses small ruts and washouts and produces false positives when tuned to attempt to detect such small features.

3.3.4. Radar Obstacle Detection

Radar sensing has several advantages for off-highway autonomous driving. It provides long-range measurements and is not normally affected by dust, rain, smoke, or darkness. Unfortunately, it also provides little information about the world. Resolution on most small antennas is limited to 1° or 2° in azimuth and 0.25 m in range. Radar scanning is generally performed in 2D sweeps with a vertical beam height of $<5°$. More narrowly focused beams are difficult to achieve and terrain height maps cannot be extracted from so wide a beam because objects of many heights are illuminated at the same time. This prevents using geometric or shape algorithms, such as those commonly used with LIDAR.

Attempts at using electromagnetic effects to gain information, such as correlating polarization with object density, have met with little success (Yamaguchi, Kajiwara & Hayashi, 1998). This leaves intensity of backscatter returns, binned by range and azimuth, as the sole identifier. Most previous systems use constant (Kaliraperumal, Lakshmanan & Kluge, 2001) or adaptive thresholding (Jiang, Wu, Wu & Sun, 2005), but achieve only marginal performance on good paved roads and are insufficient for off-highway driving. Many obstacles have surfaces that reflect energy away from the radar antenna, returning very low backscatter returns. Other objects that pose little risk to a large vehicle, such as brush, gentle inclines, and small rocks, have large radar cross sections.

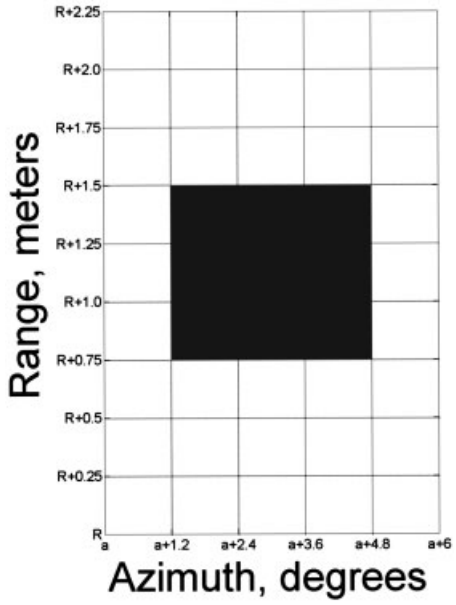

Figure 19. The kernel used in context filtering. The black pixels are multiplied by positive one and the white pixels by negative one. The center pixel is set to the sum of these values.

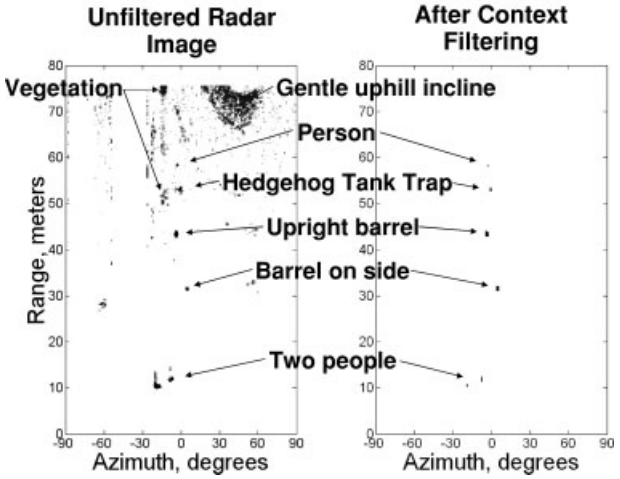

Figure 20. Example context-filtered results from a desert scene. White pixels are empty and darkness represents strength of backscatter return.

Thus, the intensity of backscatter returns is a poor measure of the risk posed by an object.

3.3.4.1. Context Filtering

The primary challenge of processing radar data is separating dangerous or interesting objects from pervasive clutter. Early experimentation with thresholding and energy filtering failed in desert environments, generating too many false positives from innocuous objects. Thus another feature was therefore required for classification.

In the desert, clutter tends to occur over wide areas. Vegetation, inclines, and rough road sections all produce backscatter returns distributed over a significant region. Conversely, obstacles, such as telephone poles, fence posts, and cars are generally isolated from each other and surrounded by road or clear dirt. Smooth ground, such as this, is specular because the low angle of radar beam incidence tends to reflect energy away from the antenna. Therefore, a potential classification feature is isolation, or the quality of a moderately low cross-section object of small footprint surrounded by clear specular areas. While this limits the class of obstacles detected, it defines an important class of obstacles that can be detected with a small rate of false positives.

To implement an algorithm exploiting this classifier, radar data are organized into a 2D image consisting of range and azimuth bins (Figure 19). A kernel consisting of two radii is convolved with this image. While the kernel is centered on a pixel, the energy between the inner and outer radii is subtracted from the energy contained within the inner radius. The value for this pixel is compared to a threshold and then reported as obstacle or not. The strength of this filter is dictated by the ratio of negative to positive space, i.e., the ratio of the two radii. The size of the inner radius determines the footprint size for which the filter is tuned. Results from a scene in the Nevada desert are presented in Figure 20.

3.3.4.2. Radar Map Hysteresis

As a vehicle drives forward, obstacles drop below the radar beam or become obscured, causing them to fade in and out of individual radar images. Thus, some form of memory is required to preserve their size and location. A simple approach is to add any newly classified obstacle pixel to a map combined with the previously reported locations of other obstacles. Due to the conversion from polar to Cartesian cost maps, the location and size of an obstacle is refined as a vehicle approaches it.

Figure 21. Two maps recorded driving along a barbed-wire fence during the 2005 Grand Challenge. The left map shows the magnified fence post at 35 m range. The right map shows the same post, this time at 15 m range. As the obstacle approaches, its size and position are refined, while obstacles behind the vehicle and out of the antenna's field of view are retained.

Hence, a modification is required to the simplistic method that stores the union of all reported obstacle pixels. If a new obstacle blob is reported and overlaps with an old blob, then the old blob is removed and replaced with a new more refined model of the obstacle location. This is implemented as a recursive algorithm that searches the neighboring pixels in Cartesian space to identify the extent of the old blob.

The results of this hysteresis are shown in Figure 21. As the vehicle approaches a fence post, the reported width is reduced from over 1.5 m to a more reasonable 0.5 m as better azimuth data become available. Obstacles that the vehicle has already passed are stored in the map even though they are not in the radar antenna's 180° field of view.

3.3.4.3. Limitations

The radar system was tested on Sandstorm and H1ghlander for over 3000 km of off-highway driving. It effectively detected the narrow range of obstacles it was designed to detect, but does not generalize beyond those. As a complement to the LIDAR processing algorithms, it makes no attempt to detect road edges or other areas of rough terrain, making radar-only off-highway navigation risky. While limited, this implementation demonstrates that, with proper scoping, RADAR can provide valuable long-range obstacle data to complement LIDAR and potentially vision-based sensing.

3.3.5. Terrain Extrapolation

Perception in high-speed outdoor navigation often suffers from incomplete data due to a combination of occlusion and vehicle motion which can cause sensors to skip over terrain. Without a method for inferring reasonable traversability values for unseen terrain, this problem can result in catastrophic fail-

Figure 22. An example of incomplete data (a) which led to significant damage (b). The red continuous line represents the preplanned path. The data show that the preplanned path is on the road edge and veers off the road, which is combined with the with the hill crest to cause the robot to leave the road.

ures, such as the example illustrated in Figure 22. To address this problem, the onboard navigation system incorporates a terrain extrapolation module (TEM). While appropriately interpreting the traversability of unsensed terrain is not new, there is very little published work in the literature (Nabbe, Kumar & Hebert, 2004).

The terrain extrapolation algorithm infers terrain costs based on three assumptions: (1) The characteristics (width, traversability) of a trail or road do not change rapidly over short distance along a path, (2) the cost of traversability can change rapidly in the direction perpendicular to the path (i.e., trails may be sharply defined), and (3) the preplanned path is parallel to the actual traversable path. Figure 23(a) gives an example of input cost data for the TEM. As a path is driven, the algorithm samples a cross section of costs perpendicular to the preplanned path, represented by the black line in Figure 23(a). Each sample is combined with a running alpha-blended average of costs from the same lateral offset. The result is a constantly evolving "learned" profile of the costs across the path. The alpha blending average filters out high-frequency terrain features, such as perpendicular fences, while allowing the TEM to adapt to new trail conditions over a few tens of meters.

The calculated trail cost profile is used to generate a "hallucinated" cost map by painting the profile along the preplanned route. The map is fused with the sensor evaluation cost maps with a low confidence. The result is a cost map with TEM data only in locations in where there are no other available cost data, as shown in Figure 23(b).

3.3.5.1. Limitations

The primary limitation of the TEM is that it requires the true traversable path to be approximately parallel to the preplanned path. Since the profile data are calculated as cross sections of the preplanned path, the profile of the preplanned path matches the profile of the traversable path only when the two paths are approximately parallel, as shown in Figure 24(a). If the preplanned and traversable paths are at significantly different angles, as shown in Figure 24(b), the profile generated by the TEM will become

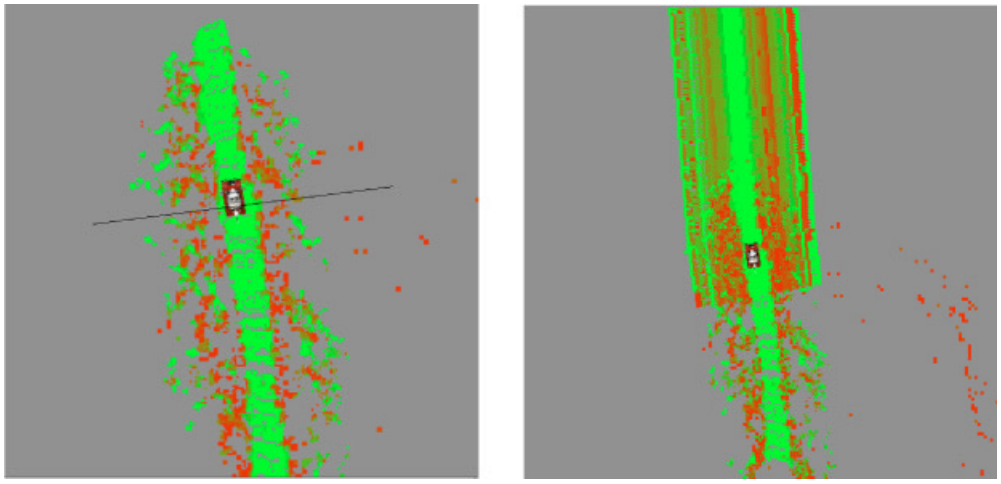

Figure 23. (a) Example of typical cost map from H1ghlander. The black line represents a cross-section sample of costs the TEM uses to produce a cost profile. (b) Same scene as (a), but with the TEM's output fused into the upper half of the cost map.

blurred as the cross sections of the traversable path do not consistently project into the cross section of the preplanned path.

3.4. Planning

The planning portion of the online navigation system is broken into a pair of modules that adjust the preplanned path based on terrain evaluation generated by the perception algorithms. The first stage, the geometric planner, adjusts the path to avoid obstacles and minimizes the cost of traversability of the terrain the robot will drive over. The speed planner operates on the output of the geometric planner and pre-emptively slows the robot for any sharp turns that may result when the geometric planner generates a plan to avoid obstacles.

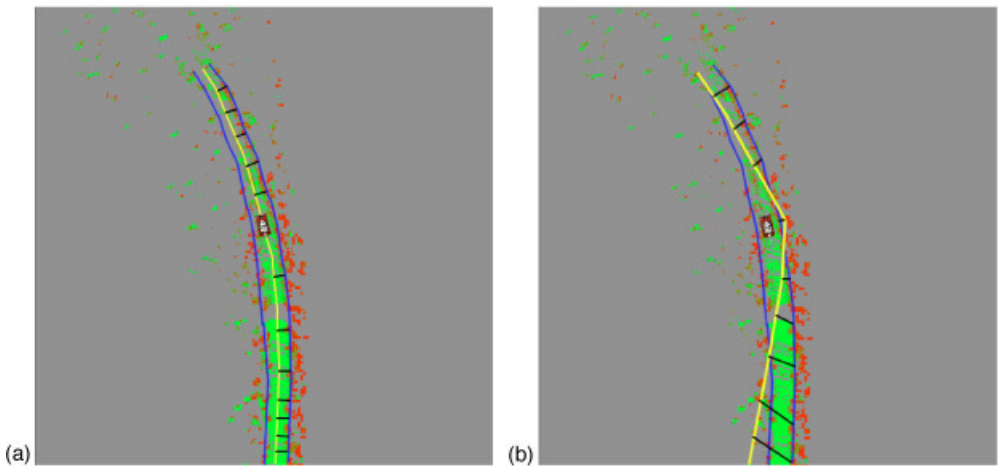

Figure 24. (a) An example of the preplanned path (light line) running approximately parallel to the traversable path (dark lines). The black lines are part of the cross section sampled by the TEM. In this case, the profile generated by the TEM produces an accurate profile of the path. (b) An example of the preplanned path not running parallel to the traversable path. In this case, the cross sections sampled do not correlate to the cross section of the traversable path.

3.4.1. Geometric Planning

Trajectory planning algorithms attempt to find an optimal path from a starting point to a goal point. There has been a tremendous amount of work in this area ranging from deterministic heuristic-based algorithms (Hart, Nilsson & Rafael, 1968; Nilsson, 1980) to randomized algorithms (Kavraki, Svestka, Latombe & Overmars, 1996; Lavalle, 1998; Lavalle & Kuffner, 1999, 2001). There has also been significant research in algorithms to set speeds and curvatures reactively (Coombs et al., 2002; Roth et al., 2005; Shimoda, Kuroda & Iagnemma, 2005). In the Grand Challenge, a prescribed route consisting of a centerline with a set of bounds was provided. The bounds and centerline did not necessarily define a road, but did constrain where a robot may travel. While the details of the terrain were unclear, the route guaranteed that there was a traversable path within the corridor. This information can be exploited to significantly reduce the computational requirements of path planning. While the approach presented here was designed for the Grand Challenge, there are many scenarios in which an autonomous robot can be given a reasonable preplan of the route it must traverse.

3.4.1.1. Conformal Search

Because the path is assumed to be traversable, the search can be limited to expansion near and in the direction of the path. A search graph is constructed relative to the preplanned path that conforms to the shape of the path and constrains the motion of the vehicle. The spacing of the graph along the path is varied to increase stability as speed increases. The graph is searched using A* and the nodes comprising the solution are connected by straight-line segments.

Possible expansion nodes are grouped in linear segments, oriented normal to the direction of travel of the path, similar to railroad ties (Figure 25). Nodes are spread with a fixed spacing across each of the segments. Each node is allowed to expand to neighboring nodes in the next segment. A node is considered to be a neighbor of another node if its lateral offset is within one step of the current node. Expansion opposing the direction of travel or within a segment is disallowed.

The cost at each node is calculated from the composite fused cost map. A rectangular window is centered on the node and aligned with the direction of travel of the path. Costs in the cost map within the window are aggregated using a weighted averaged to produce a C-space expanded estimate of cost of traversability at that node. To encourage the planner to produce paths which are shaped like the preplanned path, traversal cost to the left and right neighbors is increased by a factor $\sqrt{2}$.

Figure 25. The conformal planner operates between cells on adjacent segments normal to the preplanned path.

Each cycle, the graph is regenerated and searched using A* to produce an optimal path given the most recent sensor data. The search starts at the point closest to the vehicle on the last path output by the planner. The first few meters (proportional to speed) of the previous path are used as the starting point to generate the new path. This starting path is used to account for vehicle motion during the search.

The raw output path tends to have sharp turns; A* chooses to either maintain a fixed offset from the preplanned path or to avoid an obstacle as hard as possible. These sharp turns slow the vehicle considerably, as the speed planner will reduce speed to ensure safety. In order to remove these sharp turns, a greedy smoothing operator is applied to the path. The smoothed path is only accepted if it has a cost approximately equal to the nonsmooth path.

In most cases, the search operates quickly; faster than 20 Hz on the navigation computers. Occasionally, the search space is too complicated for the search to be completed within a reasonable amount of time. Because the vehicle is a real-time system traveling at high speed on rough terrain, planner lockup is unquestionably bad. To prevent lockup, the search times out after a 20th of a second, returning the best path found at that point. In practice, this path has been acceptably drivable.

3.4.1.2. Limitations

In situations where sharp turns are required, two of the perpendicular segments can overlap resulting in

a path that has a small knot in it. The knots rarely cause a problem but can cause the tracker to become confused and either reverse or drive poorly.

The planner does not explicitly consider speed, and the speed of the output path is assumed to be close to the speed of the preplanned path. While the speed planner will slow the output path to account for obstacle avoidance, explicitly reasoning about speed and geometry at the same time could improve performance and generate safer paths.

Since this path planner does not reason about reversing and will generate a route through obstacles if no other path is available, a second level of planning is necessary to correctly handle some scenarios. For example, the existing planner cannot correctly handle a completely blocked road.

3.4.2. Speed Planning

The speed planner is responsible for ensuring driving speeds are safe. As vehicle speed increases, dynamics become important. Speed induces side-slip (Gillespie, 1992) and can cause rollover (Diaz-Calderon & Kelly, 2005) in vehicles with a high center of gravity. Considerable research has been done to characterize vehicle performance at low speed (Golda, Iagnemma & Dubowsky, 2004) considering both roughness (Castelnovi, Arkin & Collins, 2005) and compressibility (Talukder, Manduchi, Rankin & Matthies, 2002; Talukder, Manduchi, Castano et al., 2002) for speed setting and is primarily reactive.

Maximum tractive force is limited by friction, thus speeds must be planned such that they gradually decrease as turns are approached. While obstacle avoidance and controller error can cause executed curvatures to be larger than the initial plan calls for, an estimate of maximum safe vehicle speed can be determined in advance as a function of path curvature and maximum deceleration.

3.4.2.1. Modeling

The vehicle is modeled as a point mass with rigid wheels on a flat surface. Mass is equally distributed over the wheels and does not shift. Under these assumptions, sliding occurs when static friction cannot counter longitudinal or lateral acceleration:

$$\frac{v^2}{R} < \mu g, \tag{3}$$

$$|\dot{v}| > d, \tag{4}$$

where μ is the coefficient of friction, g is gravity, v is the speed of the vehicle, d is the maximum deceleration of the vehicle, and R is the radius of curvature of the path. While d is limited by the tire-soil interaction, in practice maximum deceleration is slightly smaller because the velocity controller does not operate at the traction limit.

Numerous unmodeled dynamics including suspension effects, changes in friction, and tire stiffness cause inaccuracies in this model. In practice, it is important to incorporate a safety margin, k_{safety}, to decrease lateral acceleration and maximum deceleration:

$$\mu_{\text{eff}} = \mu k_{\text{safety}}, \tag{5}$$

$$d_{\text{eff}} = d k_{\text{safety}}, \tag{6}$$

$$v_{\text{lat}} < \sqrt{\mu_{\text{eff}} g R}. \tag{7}$$

Equation (4) can be reformulated to represent a maximum speed, v_{lat} of a point P_i relative to the following point, P_{i+1}:

$$v_{\text{lon}} = \sqrt{2 d_{\text{eff}} |P_{i+1} - P_i| + P_{i+1} v}. \tag{8}$$

Radius of curvature is defined as $ds/d\theta$ where $d\theta$ is differential heading change of the path and ds is arc length. Because the path is represented as discrete points, $d\theta$ and ds cannot be measured directly and must be estimated by $\Delta\theta$ and Δs. For a point P_k, define two headings θ^+ and θ^- as the heading of two points, $P_{k+\text{spacing}}$ and $P_{k-\text{spacing}}$

$$\Delta\theta = \theta^+ - \theta^-. \tag{9}$$

Arc length is approximated by the sum of lengths between the points between $P_{i-\text{spacing}}$ and $P_{i+\text{spacing}}$

$$\Delta s = \sum_{i=-\text{spacing}}^{\text{spacing}-1} |P_{k+i+1} - P_{k+i}|, \tag{10}$$

$$R = \Delta s / \Delta \theta. \quad (11)$$

A two-pass process is applied to determine a maximum speed at each point in the path. The first pass walks the path in the forward direction and sets a maximum speed at each point to the lower of a maximum overall speed for the vehicle and the result of Eq. (7). The second pass walks the path from the last point to the first point and limits the change in velocity so that it is constrained by Eq. (4).

The algorithm runs two ways. Before executing a path, the algorithm processes the entire path once setting speeds well beyond the sensing horizon of the vehicle. While the vehicle is driving, the algorithm runs on the output of the planner (a small subset of the overall path) allowing the vehicle to slow for unexpected obstacles.

In practice, this algorithm performs well enough to drive safely on desert trails. As the vehicle approaches turns, it smoothly decelerates to a safe speed. As the vehicle approaches obstacles, it slows as necessary to swerve around them.

3.4.2.2. Limitations

Traction is limited by the sum of two vector components: Lateral acceleration required to turn plus longitudinal acceleration of the tires (Gillespie, 1992). Because this model uses the maximum of each of these components, the maximum deceleration and coefficient of friction must be tuned for the worst case—turning while decelerating. This results in a slightly lower maximum speed through turns when the vehicle is not decelerating.

Tires are not rigid and consequently do not suddenly break away as this model suggests. In reality, as lateral acceleration increases, tires walk sideways inducing a side slip angle (Gillespie, 1992). The side slip angle is a function of many parameters including tire pressure, tire stiffness, and the coefficient of friction. When vehicles operate with high slip angles, they are hard to control. Additionally, if slip angle is not corrected for by the control layer, it will induce error.

Many effects which are functions of the terrain and environment decrease tractive force. A wheel bouncing on washboard terrain has less contact with the ground, and as a result cannot apply as much force. On side slopes and in banked turns, gravity and the "up force" generated by the curvature of the terrain changes the maximum possible speed before rollover and breakaway.

Additional system effects should also be considered. Controllers have greater error at high speed. The planner operates poorly when it has very little information. An analysis of these errors could provide a function that predicts errors at particular speeds and curvatures. A cap on these errors would determine maximum speed through turns and in straights.

3.4.3. Path Tracking

At the lowest level of the onboard navigation system, a modified conventional pure-pursuit path tracking algorithm (Amidi, 1990; Coulter, 1992) steers the vehicle. As is common, the look-ahead distance of the tracker is adjusted dynamically based on speed. The control gains are configured to provide a balance between good performance at both low speed in tight maneuvering situations, and at high speed on straight-aways and soft corners.

The basic pure pursuit algorithm works well if the output arcs are executed faithfully by the underlying vehicle controllers. Errors in the mapping between steering angle and curvature in the low level control scheme induce systematic tracking errors. For example, the steering angle sensor on Sandstorm would intermittently shift relative to its mechanical ground, this resulted in a moving zero point for the mapping between steering angle and curvature that could not be corrected by the vehicle control system. The effect of this shift was to cause the robot to track with a significant steady-state lateral offset from a desired path.

To correct for this problem, the basic pure-pursuit tracker is augmented with an integral correction function. The error term is calculated as proportional to the lateral offset between the vehicle and the path when the commanded steering angle is near zero curvature. This causes the integral term to accumulate on straight aways, but not in corners where pure-pursuit tracking would normally have significant errors. The scaled integrated curvature correction term is then added to the desired curvature generated by the basic pure-pursuit algorithm before it is passed on to the vehicle control system.

This solution worked well and effectively removed the tracking bias problems induced by the shifting steering sensor. The addition to the integral term also made the vehicle robust to some mechanical damage to the steering system that would have otherwise degraded driving performance.

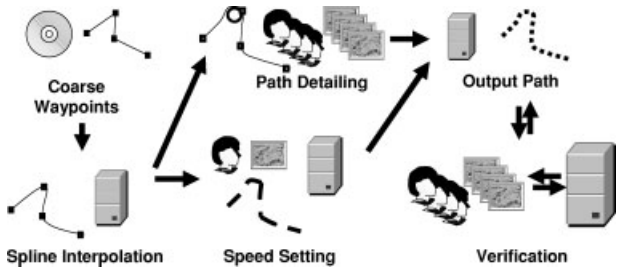

Figure 26. An illustration of the preplanning process.

4. PREPLANNING

The preplanning system creates a path, including its associated speeds and estimated elapsed time, prior to robot operation. It incorporates aspects common to mission scaled schedulers and planners for space missions (Tompkins, Stentz & Whittaker, 2004) and provides the critical input that allows the navigation system to make assumptions about the world in which it operates. Preplanning helps a robot by allowing it to anticipate and slow down for obstacles and conditions without sensing them directly. Good human drivers use similar foreknowledge to adjust radii, favor lanes, and set speeds to slow down for harsh terrain features.

As illustrated in Figure 26, the preplanning system involves the initial generation of a path using splines to smooth an initially coarse list of waypoints. The smoothed path is then modified by human editors to widen turns, and identify high risk areas in the path. In parallel, risk is assessed throughout the course and an automated speed setter uses this information to achieve a desired elapsed time. This resulting path is then output in the form of a series of fine waypoints which are used by the robot to traverse the course. As part of this process, several rounds of verification are performed, to find and remove problems with the path. Periodically, the current best path is transferred to a robot to ensure that there is always a route available, in case of some unexpected failure.

Human editors perform feature extraction that is beyond the state-of-the-art in automated image understanding. Image extraction algorithms are limited to items, such as road extraction from imagery (Harvey, McGlone, McKeown & Irvine, 2004) and object detection and delimiting (Flores-Parra, Bienvenido & Menenti, 2000) for large-scale features, such as buildings. Even though delimiting and detecting objects would be useful for the identification of underpasses, overpasses, and gates, the algorithms are only semi-automated, reducing their value for this application. Furthermore, even if these technologies were implemented, they are still unable to detect subtle obstacles, such as road washouts, which can be potentially fatal for a robot.

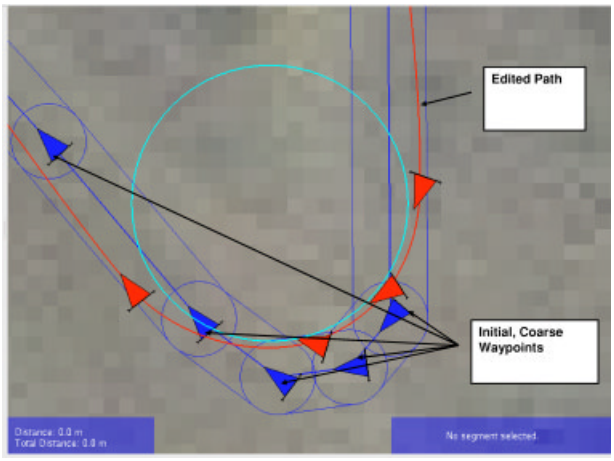

Figure 27. An example of path detailing: Adjusting a turn to achieve a minimum turning radius.

4.1. Path Editing

Path editing is a process that transforms a set of coarse waypoints and speed limits into a preplanned path with 1 m spaced waypoints. The smoothing process produces a route of curved splines from waypoint to waypoint defined by a series of control points and spline angle vectors. Human editors can alter splines by shifting control points and spline angle vectors that specify the location and orientation of a path. The generated splines are constrained to ensure C2 continuity which helps ensure a drivable path.

The human editing process removes unnecessary curvature from the smoothed path. Smooth paths are also generally faster since decreasing the amount of curvature in a path reduces concerns for dynamic rollover and side slip. Figure 27 shows an instance where an editor has widened a turning radius in a path.

4.2. Speed Setting

During preplanning, a speed setting process specifies the target speeds for an autonomous vehicle

given a target elapsed time to complete a preplanned path. Speed setting is performed by assessing the risk for a given robot to traverse a section of terrain based on available information. An automated process then uses a speed policy generated by combining the risk assessment with any speed limits imposed on the course to assign speeds to each waypoint in the path.

The risk estimation process discretizes risk into four levels (dangerous, moderate, safe, and very safe) in classifying terrain. Each risk level maps to a range of safe robot driving speeds. Risk is first assigned regionally, over multikilometer scale terrains, using general characteristic of the terrain under consideration. Once the entire route has risk assigned at a coarse level, a first-order approximation of the ease/difficulty of that route, as well as an estimate of the overall elapsed time can be generated.

In addition to classifying risk at a macrolevel, risk is also assigned to local features of importance. This processing step characterizes and slows the path down for washouts, overpasses, underpasses, and gates. In this way, human editors provide a robot with a set of "pace notes," similar to the information used by rally race drivers. These details allow a robot to take advantage of prior knowledge of the world to slow pre-emptively. This is a critical part of the process for increasing the robustness of the onboard navigation system.

4.3. Verification

The verification step helps ensure that each preplanned route is free from errors prior to a robot executing it. The verification process is performed in three ways: (1) In-line as an automated method which operates periodically while the route is edited, (2) through multiple reviews by human editors, and (3) through an automated external independent check on the final route.

4.3.1. Automated Inline Verification

The inline verification process provides human editors periodic updates of locations where the path being edited violates any constraints. The specific constraints considered are: (1) Exceeding of corridors boundaries, (2) path segments with radii tighter than a robot's turning radius, and (3) areas where the route is very narrow and warrants extra attention. Each of these potential problems is flagged for review by a human editor. These flags are then used as focal points for interpreting the path.

4.3.2. Human Review

Editors review each segment multiple times to ensure that the final route is safe. While detailing a route, each segment undergoes an initial review by editors which fixes major problems. The first review looks for any errors in the output of the automated planner, and attempts to identify areas of high risk for a robot, such as washouts. These high risk areas are then flagged, to be confirmed in a second review. A second review takes the output of the first review, and refines the route, confirming marked "flags" and adding additional "flags" for any high risk areas missed in the first review. The expectation is that after completion of the second review, there will be no need for additional editing of the geometry of the route. In the third and fourth reviews, the main focus is to verify that all problems identified by the automated inline verification process have been cleared, as well as to confirm that any problems identified by the automated external verification algorithm are addressed.

4.3.3. Automated External Verification

An automated external verification process operates on the final output to the robot and checks heading changes, turning radius, speeds, and boundary violations. In addition, the verification process outputs warnings where there are areas of high slopes and sections of narrow corridors along the path. These warnings are used to identify areas for the human editors where extra care should be used. The verification process also produces a number of strategic route statistics, such as a speed histogram for time and distance, a slope histogram, and a path width histogram. These statistics are used in determining the target elapsed time for the route and in estimating the risk for the route. This process is repeated several times as the path detailing progresses until the route is deemed safe for the robots to use.

5. TESTING AND SYSTEM PERFORMANCE

To succeed in the Grand Challenge, robots must demonstrate both performance and reliability. To achieve

Figure 28. H1ghlander (and Sandstorm) were tested extensively on desert terrain.

these two requirements, the team identified that a system of tests and performance milestones were required. Performance milestones were set to ensure driving skills and reliability would meet near-term program goals, such as passing the site visit demonstration, winning pole position at the National Qualifying Event (NQE) and completing the Grand Challenge. Examples of performance milestones included shakedown cruises, NQE skills tests, and race length autonomous runs at a race pace and in a race environment. In addition to milestones, the team developed test methods that enabled quantitative evaluation of the effects of changes to the robots' hardware and software on performance (see Figure 28).

5.1. Controlled Tests

The impetus for a controlled test methodology was to monitor development progress and measure driving skill. The team also used the tests to evaluate the effects of changes in hardware and software on the robots' overall ability to drive. The ability to drive was defined as three major skills. The first skill was the ability of the robots to follow a preplanned path based on position sensing only. The second skill was the ability of the robots to track a preplanned path while assisted by perception sensors. The third skill was the ability of the robots to dynamically make significant modifications to a preplanned path to avoid sensed obstacles.

5.1.1. Test Formulation

There are few, if any, documented standardized tests to measure autonomous driving skills at speed. A review of technical reports of DARPA Grand Challenge 2005 finishers Stanford Racing Team (Stanford Racing Team, 2005), Team Gray (Trepagnierl, Kinney, Nagel, Dooner & Pearce, 2005) and Team Terramax (Oshkosh Truck Corp., 2005) found that they all placed great value on testing but did not mention specific tests to measure driving skill. While a method for characterizing tracking performance has been described (Roth & Batavia, 2002), the authors are unaware of literature that describes a standardized process for evaluating perception based navigation.

The literature on the subject of automotive dynamic testing includes International Organization for Standardization standard ISO-3888-1, Passenger cars—Test track for a severe lane-change maneuver Part 1—Double lane-change (International Standards Organization, 1999) (Figure 29). This test was designed as a means to subjectively evaluate vehicle dynamic performance. The test is subjective because it only quantifies a small part of a vehicle's handling characteristics and is highly dependent on the input from the driver. This dependence on driver skill is what makes the test attractive for adaptation to autonomous ground vehicle driving skill testing.

The original ISO-3888-1 course was modified,

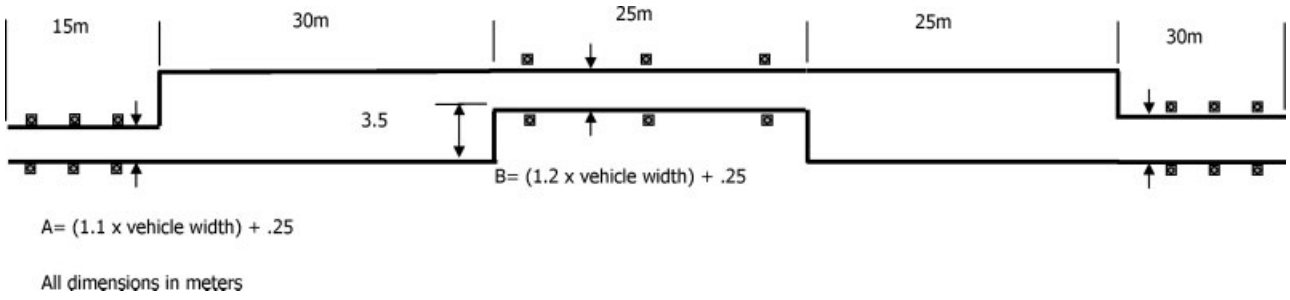

Figure 29. ISO-3888-1 test track for a severe lane change maneuver.

Table V. Test steps for autonomous ground vehicle path tracking skill assessment.

1.	Create a route file through the test course.
2.	Create a path definition file for the route created in step 1 setting the corridor width slightly wider than the test track's lane width and the speed to a constant (e.g., 5 meters/sec). The path definition file must include an area before the test course begins for the robot to achieve the required constant velocity.
3.	Load the path definition file into the robot.
4.	Command the robot to drive the route described in the path definition file.
5.	Record the time the robot is on the test track entry to exit.
6.	Record the number of times the robot touches or exits the test track's boundaries.
7.	Repeat steps 2 through 6 increasing the speed by an incremental value (e.g., 2 meters/second) until the robot can no longer successfully traverse the course or the operation is deemed to be unsafe. Multiple runs at each speed increment are required to demonstrate consistency.

adding 1.5 meter to lane width in all sections to account for nominal pose estimator and tracking errors. This additional lane width enabled a quantitative measurement of performance considering total system error. Table V describes the basic steps used in the autonomous ground vehicle path tracking skills assessment.

5.1.2. Blind Path Tracking Test

The blind path tracking test uses the modified ISO-3888-1 to perform a quantitative evaluation of path tracking performance. A route file for this test is created by recording a smooth, "best line" route through the test course. The recorded path is then transferred to the robot and used as the baseline path to be tracked. Figure 30 is a graphical representation of the path definition file used in the blind and perception assisted path tracking tests. When conducting the blind path tracking test, the robot is configured such that only the pose sensor is considered in path planning. The absence of all perception sensor input limits path tracking error to that induced from the pose sensor, path tracking algorithm, and drive by wire actuation control errors.

5.1.3. Perception Assisted Path Tracking Test

The perception assisted tracking test is conducted using the same path definition file used in the blind path tracking test and shown in Figure 30. At the entrance to each lane segment, a perpendicular boundary wall has been added to the original ISO-3888-1 test track. These boundary walls are intended to ensure a robot will plan a route through the desired lanes. While performing this test, robots are configured to use all of their perception and pose sensor data. The test measures driving skill when given a nominal path through an area constrained by boundary obstacles.

5.1.4. Perception Planning Test

The perception planning test is conducted on the same modified ISO-3888-1 test track but uses a route file that follows the center line of the test track (see

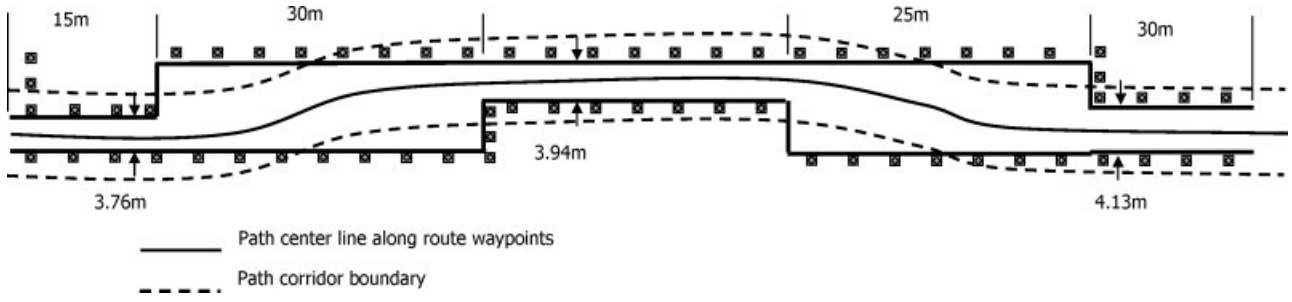

Figure 30. Bind and perception assisted path tracking route through modified ISO-3888-1 test track.

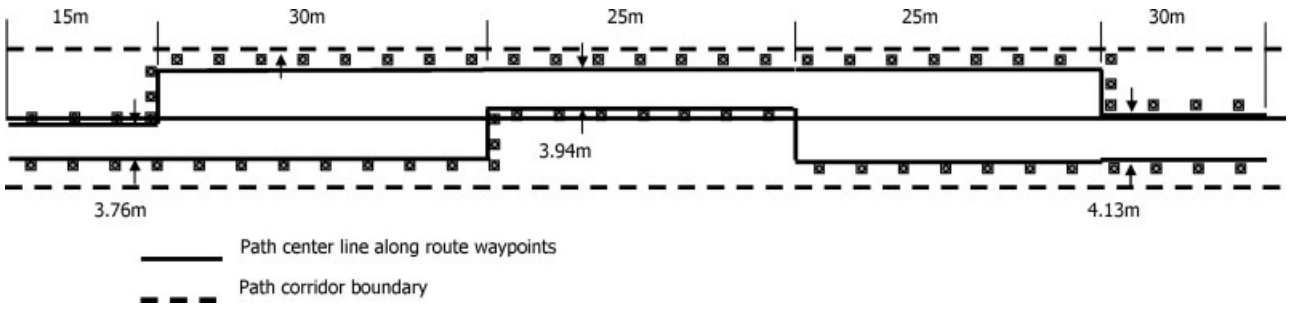

Figure 31. Perception planning route through the modified ISO-3888-1 test track.

Figure 31). This test measures an autonomous ground vehicle's ability to significantly modify the preplanned path based on perception.

5.1.5. Test Execution

Field tests were conducted at a brown site in Pittsburgh, Pennsylvania and at the Nevada Automotive Test Center (NATC) in Silver Springs, Nevada. While testing at the Pittsburgh site, small cardboard boxes wrapped in plastic bags were used as lane boundaries. As the perception capability improved, cones were substituted for lane boundaries and obstacles. Figure 32 shows Sandstorm and H1ghlander operating on the test tracks in Pittsburgh and Nevada.

During development and testing a full battery of tests was performed on the modified ISO-3888-1 test track on eight separate occasions. Test personnel included the test conductor, robot operator, chase car driver, and timers. In general, a complete set of blind path tracking, perception-assisted tracking, and perception planning tests were performed during each session. Table VI is an example of data collected during a typical test session on the modified ISO-3888-1 test tracks.

Sessions on the modified ISO-3888-1 track generally started with the blind path tracking test at an initial speed of 5 m per second. The team would execute a minimum of three runs then increase the speed by 2 m per second. At each speed increment, three runs were executed. Speed was increased until the vehicle left the course and had to be stopped via the emergency stop link or the test team deemed operations at higher speeds were unnecessary. The blind path tracking test was never conducted at speeds above 13 m per second. As confidence in the robot's blind tracking ability increased, the number of blind tracking tests decreased, and eventually were not included in the test routine. The test was held in reserve for regression testing after hardware or software changes were made to a robot that would affect the basic path tracking.

Figure 32. Sandstorm (a) and H1ghlander (b) on the Pittsburgh and NATC test courses.

Table VI. Example data collected during testing.

Test No.	TOD	Type	200 m (s)	Speed (m/s)	Notes
1	2:35	BT-5	40.98	4.88	Cone 9L brushed
2	2:40	BT-5	41.20	4.85	Cone 4L brushed
3	2:44	BT-5	41.13	4.86	Good
4	2:49	BT-7	29.20	6.85	Cone 4L hit hard
5	2:52	BT-7	29.45	6.79	Cone 4L hit hard
6	3:09	BT-7	31.70	6.31	Cone 4L hit more hard
7	3:12	BT-7	31.67	6.32	Cone 4L hit more hard
8	3:15	BT-9	ABORT		E-stop because robot leaving course at 4L
9	3:40	PT-5	45.58	4.39	Good
10	3:58	PT-5	45.26	4.42	Good
11	4:04	PT-5	43.64	4.58	Good
12	4:09	PT-7	36.67	5.45	Good
13	4:11	PT-7	37.29	5.36	Good
14	4:15	PT-7	38.70	5.17	Cone 7R hit
15	4:17	PT-9	29.70	6.73	Cone 7R hit
16	4:21	PT-9	31.27	6.40	Cone 7R hit
17	4:25	PP-5	45.89	4.36	Center 3rd wall hit
18	4:28	PP-7	39.72	5.04	Corner 1st wall brushed, 3rd wall corner crushed
19	4:30	PP-9	37.60	5.32	Corner 1st wall, brushed, 3rd wall corner crushed
20	4:32	PP-9	N/A		2nd outer and center wall hit, 3rd corner crushed
21	4:37	PP-9	33.61	5.95	3rd wall corner, 2 wall center punch & corner, 1st wall box crunched

Like the blind tracking test, the initial speed for perception tracking test was 5 m per second. Multiple runs at the slower speeds were eventually found to be unnecessary. As in blind path tracking, incremental speed changes were of 2 m per second, and the maximum speed was 13 m per second. As confidence in the perception sensing systems' ability to correctly sense and localize obstacles increased, emphasis on conducting perception tracking was diminished.

The perception planning test was performed in the same way as the perception tracking test, with the exception of using a different path.

5.1.6. Evaluation

The initial blind tracking test of Sandstorm (see Figure 33) conducted on June 17, 2005, revealed that the robot did not have a robust path tracking ability. The observed quality of driving was poor with significant overshoot when cornering. The path tracking control algorithm was modified, adding an integral term, and when the test was repeated on August 17, 2005, performance was notably improved. The team was satisfied with Sandstorm's blind tracking performance at this point and did not conduct the test on Sandstorm again. Similar performance improvement trends were observed with H1ghlander as well.

Analysis of the data collected during the perception tracking and planning tests is inconclusive of overall performance gains due to changing hardware and software configurations, but it did help detect system anomalies. As an example, performance of the vehicles was observed to decline in September after the short-range sensors were removed and replaced during addition of a sensor washing system. This decline was directly attributable to the lack of adequate calibration of the sensors for object localization.

5.1.7. Perspectives on Controlled Tests

The blind tracking, perception tracking, and perception planning tests are an effective tool for measuring autonomous ground vehicle driving skill. The tests are relatively easy to set up and inexpensive to conduct. While straight forward, the tests are an effective means to evaluate hardware and software configuration changes.

The blind tracking test is an excellent tool for measuring an autonomous ground vehicle's ability

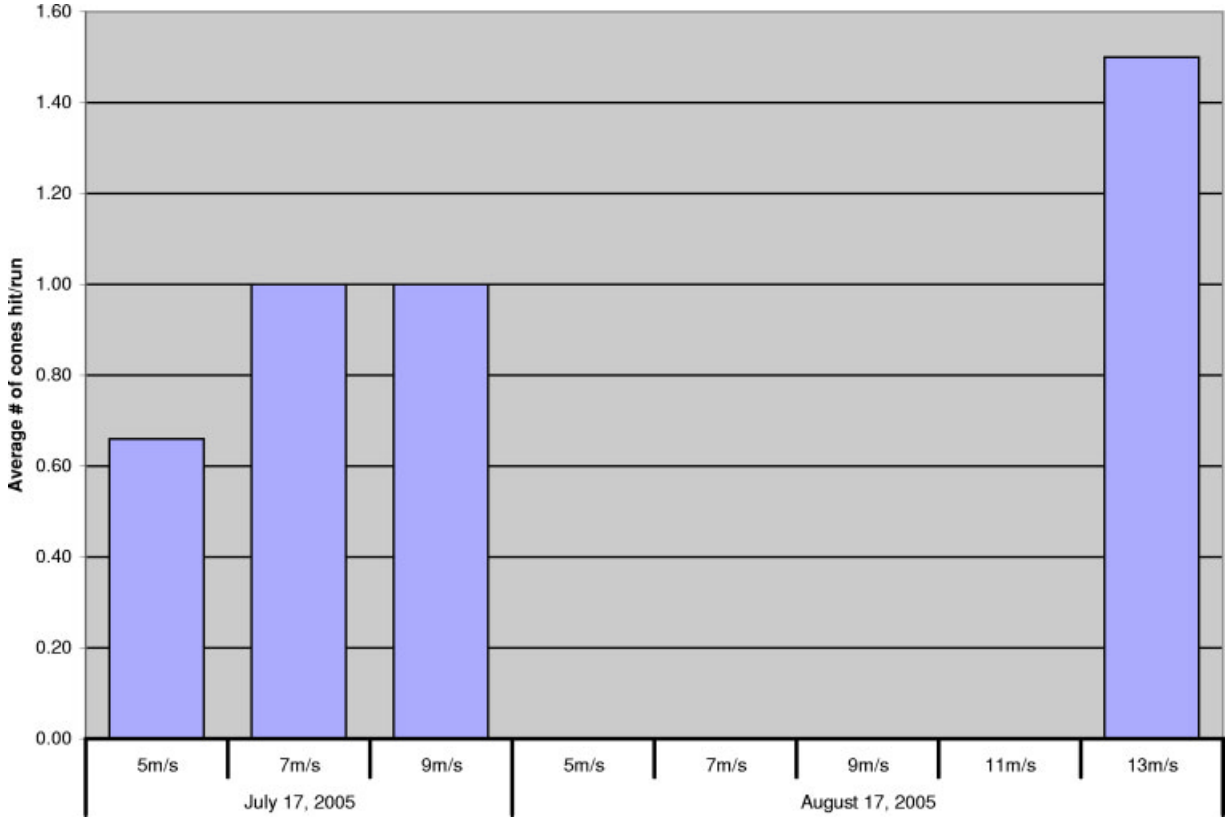

Figure 33. Blind tracking performance for Sandstorm.

to blindly follow waypoints and is broadly applicable to all automotive and truck class autonomous ground vehicles.

The perception tracking test is a good tool to measure the effects of perception on path tracking and also as a subjective qualitative assessment of driving quality. The perception planning test is a good tool to measure the effectiveness of an autonomous ground vehicle's ability to dynamically adapt to obstacles impeding the preplanned path. The perception planning test is limited; while it measures a robot's ability to rapidly adapt to obstacles, it does not measure if a vehicle reacts in a smooth manner. For example, if a human is driving and sees an obstacle in its path, they will generally react as early and smoothly as possible to change their trajectory to avoid the obstacle. The perception planning test could be adapted for this purpose by elongating the length of the track segments between lane change barriers.

5.2. End-to-End Tests

The Red Team was deployed to Nevada in late August of 2005. Once there, the team began conducting a series end-to-end system tests designed to simulate race day conditions. These tests followed the following process.

- System test team identifies a route and creates a route data definition file (RDDF) (this could take several days to complete).
- The RDDF is delivered to the preplanning team who have 1 h and 45 min to create a path file (ideally, this was accomplished on the morning of the systems test).
- The path file is reviewed by the field team to ensure that it is safe and viable.
- A system test team provides the path file to the robot operators who have 15 min to load and launch the robot (this step was timed in

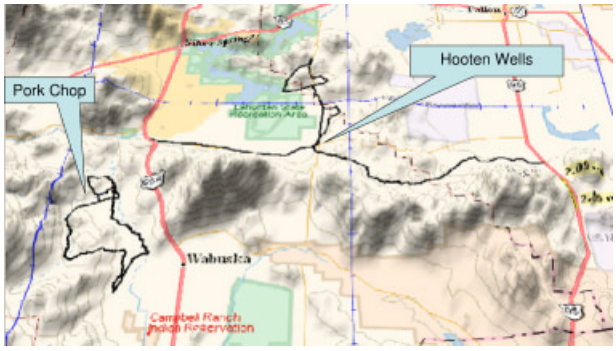

Figure 34. The "Pork Chop" route is a 48 km loop while the "Hooten Wells" route is 85 km each way.

order to validate the operators' ability to reliably launch the robot in a timely manner).

- Robots were then chased and performance monitored until either run completion or intervention.

Identifying routes that were race length including a wide collection of terrain, free from undue risk, proved to be a formidable task. No nonlooping race length route was identified near the test area. Figure 34 shows two of the routes used for systems tests. The "Pork Chop" route is a 48 km loop featuring pavement, gravel road, dry lake bed, sand, wash out, dirt road/trail, cattle guard, gates, parallel fences, high-voltage power lines, rail road crossings and elevations ranging from 4300 to 4750 feet. The "Hooten Wells Extended" route is 85 km one way. This route was looped by adding tight circle turns at both ends. The Hooten Wells route features a gravel road, dry lake bed, large wash outs, dirt road/trail, cattle guard, gates, parallel fences, high-voltage power lines, a canyon, and elevations ranging from 4000 to 4700 feet.

During tests along these routes Sandstorm and H1ghlander each drove over 1600 kilometers autonomously. Each robot completed challenge length runs at a race pace on routes more difficult than the 2005 DARPA Grand Challenge race route. Figure 35 shows the daily total number of meters driven during testing at the NATC. The figure shows dates of the three significant failures incurred during testing: H1ghlander sheering off its front right wheel, Sandstorm being "clothes lined" by a tree and H1ghlander rolling.

5.3. Test Conclusions

Over the course of this development, the technology readiness level (TRL) (Mankins, 1995) of the robots presented in this paper improved from TRL 3 (analytical or experimental characteristic proof of concept) in the summer of 2003 to TRL 5 (Technology component demonstration in a relevant environment). This change in readiness was driven by the rigorous test program. In order to achieve TRL 9 (actual technology system qualified), a much wider set of systems tests must be developed. Although the testing described above effectively prepared Sandstorm and H1ghlander to compete in the DARPA Grand Challenge, it is not adequate for fielding an autonomous ground vehicle for everyday use.

Standardized tests must be developed that measure a robot's ability to sense and accurately localize obstacles of varying size. These tests should account for differing perception sensing modes. Standard tests that measure an autonomous vehicle's ability to safely and reliably interact with other vehicles and humans are needed. These tests and others are required in order to move autonomous ground vehicles from technological curiosities to common tools used by people everywhere.

6. THE GRAND CHALLENGE

The 2005 DARPA Grand Challenge began at 6:40 am on October 8, 2005 with H1ghander and Sandstorm departing the starting chutes first and third respectively. Both robots completed the challenge, finishing in third and second place (Figure 36). Figure 37 illustrates the race day route. The route consists primarily of wide straight dirt roads. There are a few technical areas including three underpasses and a narrow winding descent through Beer Bottle pass. Overall, the route is less difficult than the system test courses that Sandstorm and H1ghlander had operated on prior to the Grand Challenge.

6.1. Strategy

To increase the chance of success, the two robots were run with different speed ranges for each of the risk levels used by the preplanning system. Table VII describes the speeds used for each risk level. H1ghlander ran at near full speed, while Sandstorm operated with a more conservative speed plan. Both

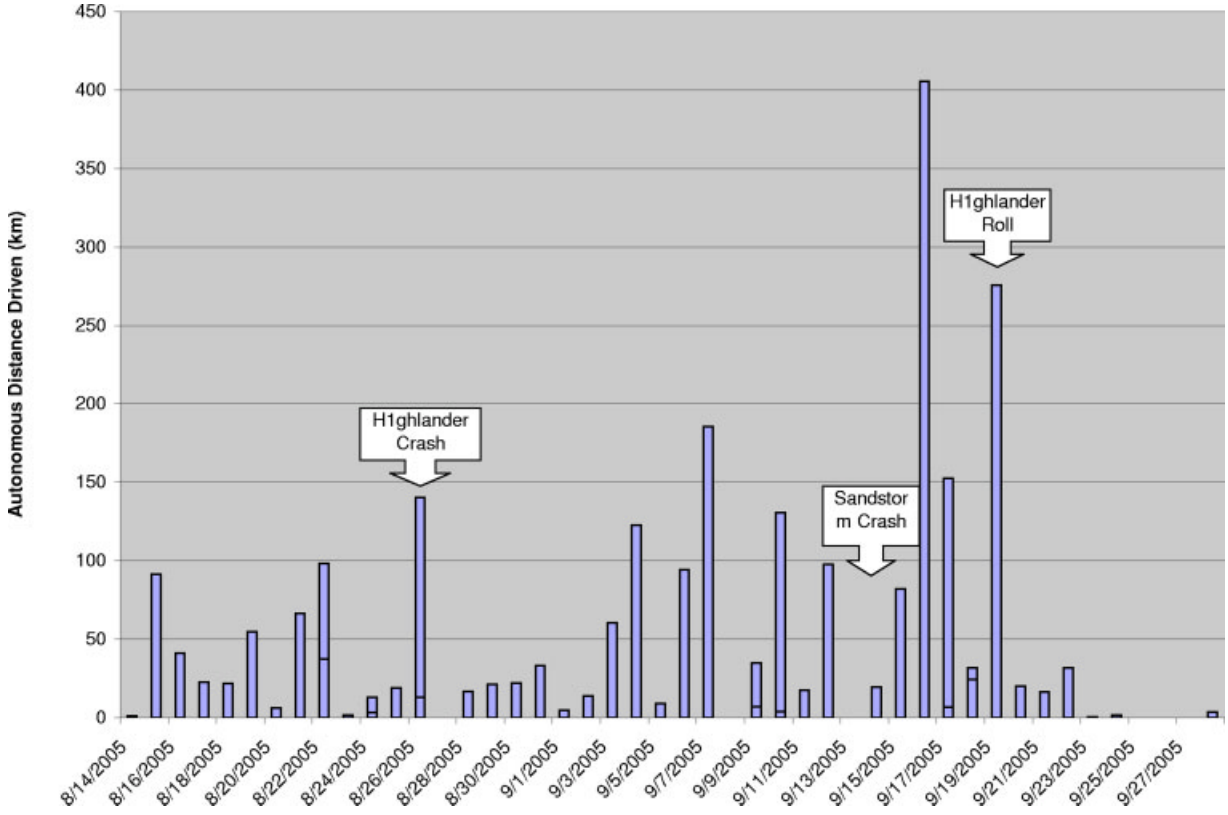

Figure 35. Daily autonomous driving distance during testing at NATC.

speed policies were within the operational ranges for which the robots had been tested. The speed policies resulted in preplanned expected completion times of 6 h and 19 min for H1ghlander, and 7 h and 1 min for Sandstorm.

The dual speed strategy was selected to increase the likelihood of at least one Red Team robot correctly dealing with some challenging unforeseen obstacle on the Grand Challenge route. While the strategy was successful in that both robots completed the challenge, it limited Sandstorm below its ability and, in retrospect, prevented it from winning the Grand Challenge.

6.2. Sandstorm

Sandstorm completed the 212 km course in approximately 7 h and 4 min to finish, in second place, 11 min behind the winning team from Stanford. At approximately 6:50 am, Sandstorm launched on time and without incident. Upon departure, all mechanical, sensing, and navigation systems were operating nominally. In general, Sandstorm drove the route well. It completed the course within 1% of the preplanned time, and came within 11 min of winning the Grand Challenge. Sandstorm drove cleanly through most of the course, only touching an obstacle during the tricky Beer Bottle pass descent.

Sandstorm cleanly navigated through three underpasses. Both Underpass 2 and 3 were sufficiently long and constrained that the LIDAR terrain evaluation algorithm failed to detect the walls (see Figure 38). Fortunately, the redundancy in perception algorithms made the navigation system robust to this failure and Sandstorm cleanly navigated all three tunnels without contact.

The only contact Sandstorm made with an obstacle during the Grand Challenge occurred during the descent through Beer Bottle pass. It is impossible to perform a complete reconstruction of the incident

Figure 36. Sandstorm (a) and H1ghlander (b) cross the Grand Challenge finish line in second and third place.

since the log file containing the pose of the gimbal is corrupt and the documentation camera stopped operating prior to entering the canyon. Figure 39 shows a cost map constructed from only the shoulder, short-range LIDARs, and thus provides an incomplete picture of what the robot saw. It is likely that the robot attempted to cut close to the wall to avoid the incorrectly evaluated terrain to the left of the trail. Despite this minor contact, Sandstorm emerged from the canyon relatively unscathed.

Sandstorm drove well and completed the grand challenge course but was on average 0.25 m/s too slow to claim victory.

6.3. H1ghlander

H1ghlander completed the challenge in 7 h and 14 min, 55 min longer than its intended completion time. This failure to achieve intended speeds and elapsed time was unprecedented in H1ghlander's testing history.

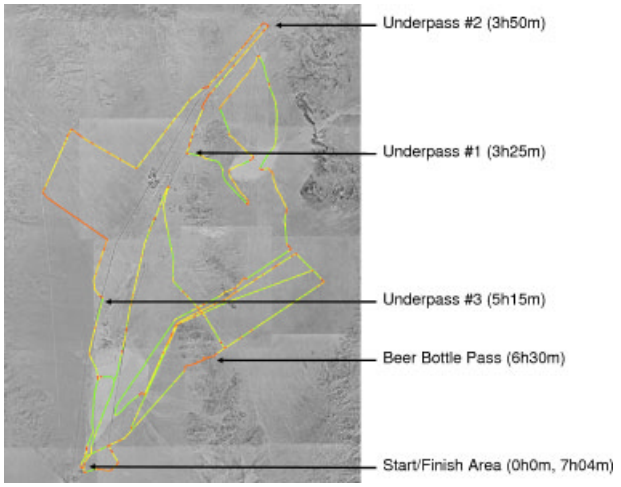

Figure 37. The 2005 Grand Challenge Route with the approximate times at which Sandstorm encountered challenging terrain.

6.3.1. Engine Trouble

Figure 40 shows predicted progress plotted versus actual progress. The upper curve shows the actual time to reach a given distance, while the lower shows predicted performance. Figure 40(a) shows a run of a very challenging course in northern Nevada. In this figure, the predicted and actual curves are almost indistinguishable. Figure 40(b) shows performance during the Grand Challenge course. Early in the race (27 km), the predicted progress

Table VII. Risk level to speed range mapping during the Grand Challenge.

Risk level	Speed ranges for Sandstorm (m/s)	Speed ranges for H1ghlander (m/s)
Dangerous	5	5
Moderate	7.5	7–9
Safe	10–10.5	10–12
Very Safe	12	13–13.5

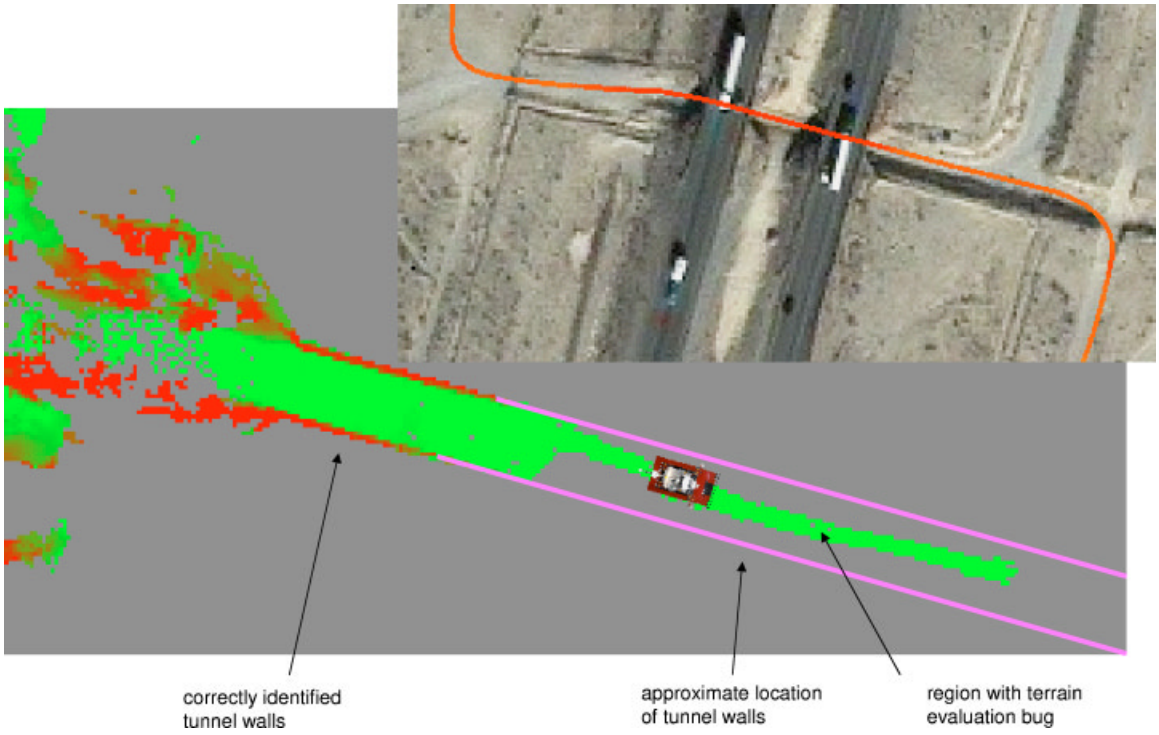

Figure 38. The third underpass and sensor data illustrating the terrain evaluation bug.

separates from the actual progress. The actual progress continues to separate throughout the race.

Figure 41 compares velocity controller performance on various grades during the race and during the last long distance traverse before the race. The prerace plot shows a slight decline in performance at higher grades while the in-race plot shows a significant decline on all grades.

Analysis of velocity performance shows seven locations where H1ghlander came to a stop. Two of these stops were on hills where the vehicle rolled backward several times before being able to crest the hills. Significantly degraded performance is first noticeable after 27 km and continues for the entire challenge. Onboard audio indicates that the engine was stalling regularly while H1ghlander drove. Immediately following the race, the engine idle speed was oscillating violently. Data were logged while this phenomenon was occurring; but since the race, the problem has not reoccurred. Samples of engine and transmission fluids show no anomalies and the onboard engine diagnostics showed no faults during and after the race. At this time, the cause for this engine problem remains unexplained.

6.3.2. Gimbal Failure

H1ghlander's gimbal stopped responding to commands 87 km into the race, failing after a hard left turn. Examination of vehicle speed indicates that this is an area where H1ghlander was having engine trouble.

The gimbal is sensitive to power fluctuations, which may have been the cause of its failure. While H1ghlander's power is typically stable, the engine conditions discussed above may have caused significant power generation problems. When the engine's rpms are low, the generator stops producing power and the power system switches to using batteries. This condition is rare, as engine idle speed is set to be significantly higher than the generator cutoff speed. Power system switching was tested repeatedly early in the development of H1ghlander, and had not shown any problems leading up to the race.

It is conjectured that while the engine was not running normally, the generator shut off temporarily, causing a brown out in the gimbal power system. The gimbal stopped responding to commands and never recovered. Normally, when processes fail

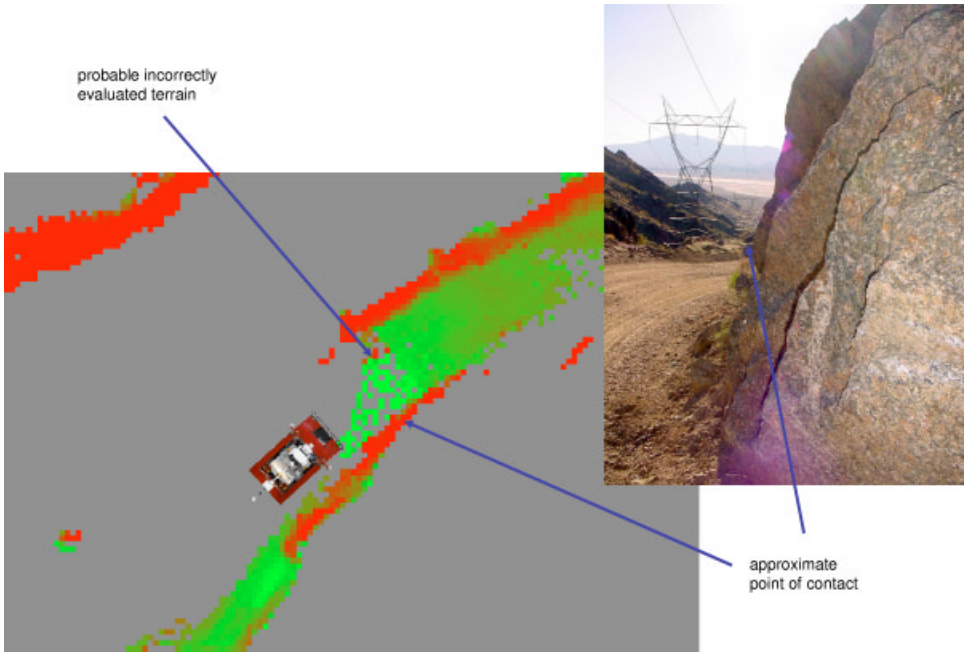

Figure 39. Sensor map illustrating the point of contact.

to respond, they are restarted. In this case, the gimbal control computer would have also browned out and as there is no capability to restart failed computers, the restart system did not attempt to restart the gimbal processes.

While this hypothesis seems to be the most logical explanation for the gimbal failure, there are insufficient data to determine the root cause of failure as power diagnostics are only logged into a short rolling buffer. Under normal testing conditions, system failures are examined immediately and information about power and state of components involved in the failure is recorded. Due to the nature of the race, the vehicle could not be stopped for examination, thus these data were lost.

Despite these two failures, H1ghlander completed the Grand Challenge, illustrating the ability of the robot to survive a number of significant faults.

7. CONCLUSIONS

Through the application of simple well-implemented ideas, it is possible to achieve robust high-speed desert navigation. At each point in the development process, we attempted to use the simplest possible approach, be it the first-order approximations of physical constraints in the speed planning algorithm or the expectation based map merging algorithm. The key to success was performing testing and analysis to understand where these approaches would fail, and determining whether a more complex solution was warranted.

The robustness of this approach was clearly demonstrated by Sandstorm and H1ghlander's performances at the 2005 Grand Challenge. Despite a failing engine and the loss of a perception sensor, both robots completed the course within 20 min of the winning time. Without the careful design and implementation of the navigation system, and the selection of a robust vehicle chassis, this feat would not have been possible.

7.1. Next Steps

While the DARPA Grand Challenge represented a significant step forward for high-speed navigation in desert terrain, the robotic navigation problem is far from solved. Before it will be possible to deploy fully-autonomous robots, techniques for dealing

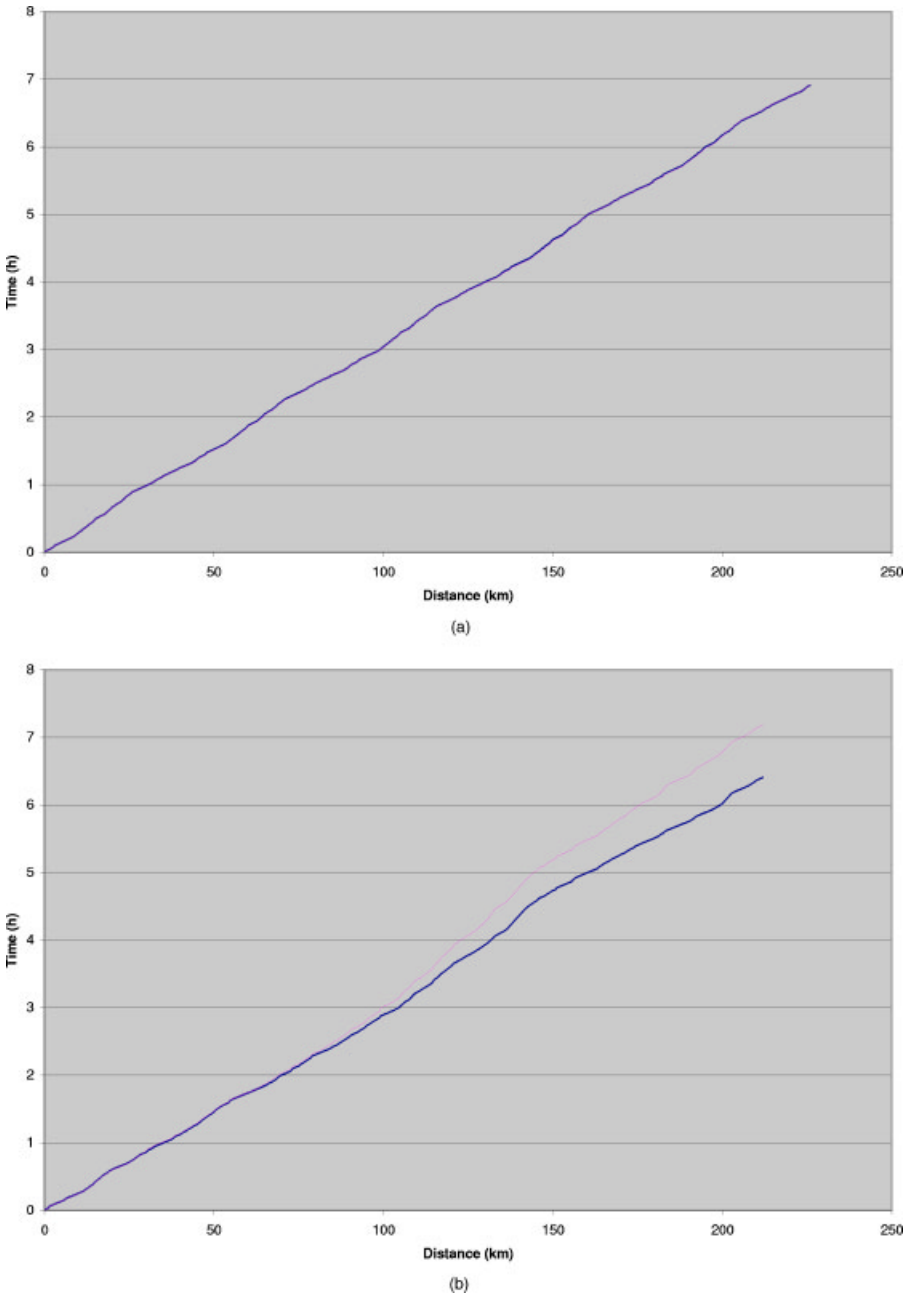

Figure 40. A comparison of H1ghlander's actual and preplanned distance traveled for (a) a representative test and (b) during the Grand Challenge.

with dynamic environments must be developed. The Grand Challenge was carefully designed to keep the competitors separated. Passing in the Challenge was constrained such that slower vehicles were stopped well before, and throughout the time, faster vehicles encountered them. While this was reasonable for the Grand Challenge, it is not a viable way for robots to deal with traffic in general.

While the desert provides numerous challenges, the terrain is relatively straightforward to model.

Most desert terrain is well characterized by its geometry since there is relatively little vegetation. This simplifies the perception and terrain evaluation tasks. The same cannot be said for jungles, forests, or even farmland. In these environments, a purely geometric understanding of the world would be misleading since there is vegetation that can and must be driven over and through. Completing the same challenge in heavily vegetated terrain would be a tremendous next step.

The Grand Challenge forced teams to develop reliable near turn-key solutions by requiring a long duration performance on a specific day. The race format meant it was not possible to develop a technology that worked some of the time, it was necessary for the robots and supporting systems to work when called on. While this level of reliability and readiness was met, it was achieved with large teams with very specialized knowledge and training. To move this technology from a compelling demonstration to a production ready capability will require significant effort to both harden and quantify performance.

7.2. Lessons for Development of Autonomous Vehicles

While Sandstorm and H1ghlander performed well, and the experience of developing and testing the robots was rewarding, there is much room for improvement in both the performance and the process used to develop them. Over the course of this research, we learned several lessons:

- *System testing is essential but results can be misleading.* Testing was critical to the success of Sandstorm and H1ghlander, but at times also provided a false sense of confidence. Because of the tight development and testing schedule, component testing was often short and incomplete, due to an emphasis on integrated testing. Since the overall system was relatively robust to sensor failures and bad data, it was possible for intermittent and subtle problems to go undetected for long periods of time. A combination of better unit/component testing or a more in-depth analytical evaluation of developing algorithms would have reduced this problem.
- *Reliability is critical, and can be achieved.* One of the keys to success in the Grand Challenge was balancing innovation with engineering. We used a programmatic approach that effectively annealed ideas to strike this balance. In the first phase, we cultivated a broad set of potentially good ideas, hoping to uncover those that would be the keys to success. In the second phase, we focused on these ideas while achieving internally and externally imposed milestones (e.g., complete a 320 km traverse, or perform obstacle avoidance with a given sensor). These milestones allowed us to judge the viability of competing technical approaches while working to also increase reliability. In the third and final phase, we accepted only ideas that were required to fix problems with the existing system. We used a strict process to report any problems identified during testing and then worked to quickly rectify them in a way that prevented recurrence of the same fault. During this final phase, the robots began to realize their potential for reliable operation.
- *Use commercial off-the-shelf (COTS) components, but only with support.* A common mantra in developing new systems is to buy not build, and in general this is true, but buyer beware. Without support, COTS components can be worse than in-house custom built components. With in-house built components, there is at least someone who designed and built the system who can debug it. An unsupported COTS component that behaves unexpectedly may force a significant redesign of a subsystem, potentially incurring more cost and time than if the component had been in-house custom engineered.
- *Know the problem.* Much of the technical approach described in this paper was excessive given the final form of the Grand Challenge. The groomed roads and carefully detailed route provided by the organizers greatly reduced two of the competitive advantages (namely the H1 & HMMWV chassis and the preplanning system) applied by the team. Furthermore, the team put an excess of wear-and-tear on the vehicles during testing operating on more rugged terrain than that encountered during the challenge. Had the final race conditions been known ahead of time, it would have been possible to shed a significant amount of technical complexity.

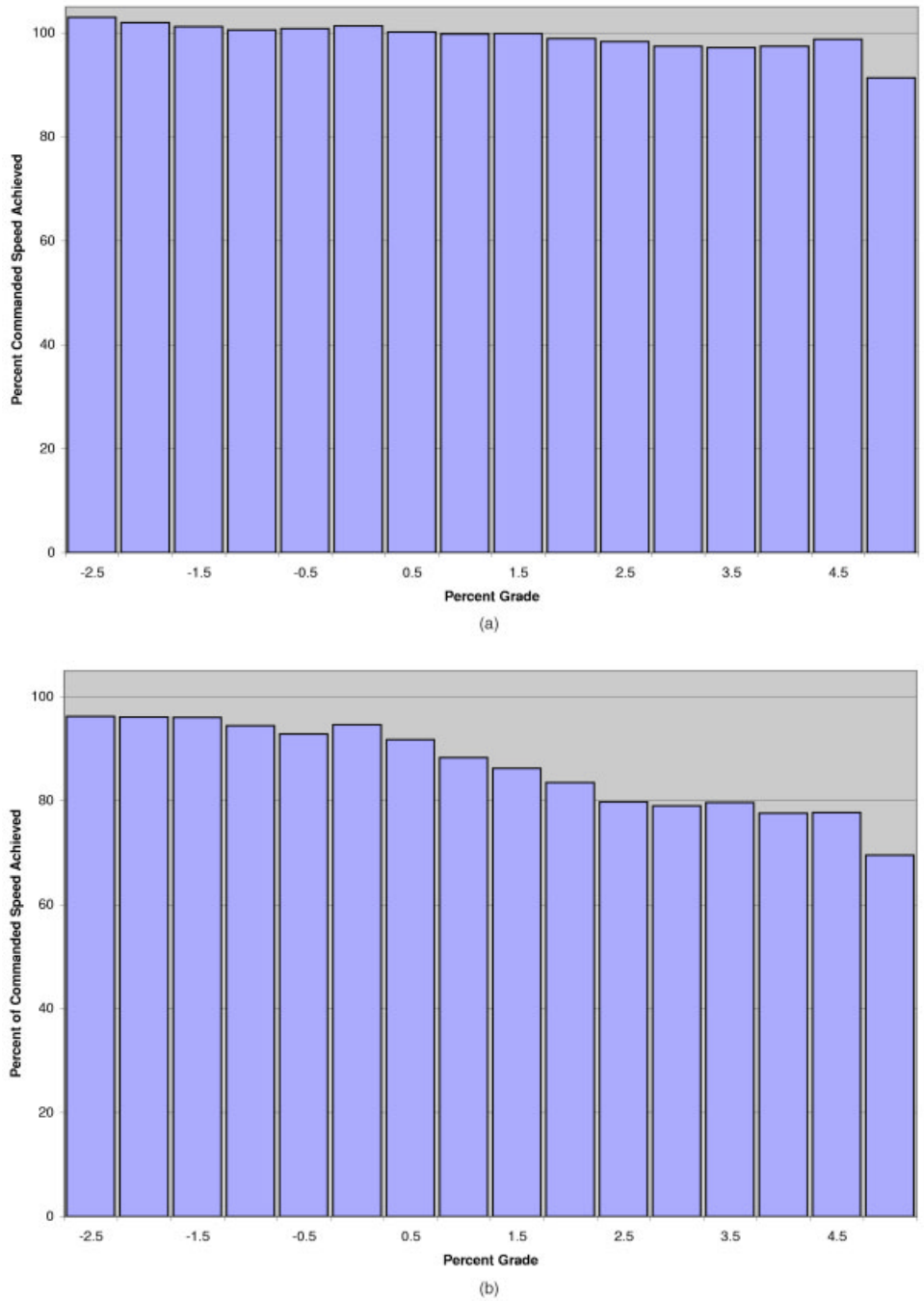

Figure 41. A comparison of H1ghlander's ability to maintain commanded speed between (a) testing and (b) performance during the Grand Challenge.

- *Correctly scoping a problem is a key to success.* One of the programmatic successes was a process to cut technical ideas if they were not progressing or showing results. This helped keep the team focused on the overall goal of completing the Grand Challenge rather than chasing after interesting ideas that were irrelevant to the task at hand. Without this focus,

we would not have been able to achieve the reliability and robustness necessary to complete the Grand Challenge.

ACKNOWLEDGMENTS

This work would not have been possible without the tremendous support of our team sponsors including Caterpillar, SAIC, Boeing, Intel, HD Systems, Harris, Applanix, Chevron, Phillips, AMGeneral, TTTech, and the numerous other contributors.

REFERENCES

Amidi, O. (1990). Integrated mobile robot control (Tech. Rep. CMU-RI-TR-90-17). Pittsburgh, PA: Carnegie Mellon University, Robotics Institute.

Batavia, P., & Singh, S. (2002, May). Obstacle detection in smooth high curvature terrain. Paper presented at the IEEE International Conference on Robotics and Automation (ICRA 2002), Washington, D.C.

Bellutta, P., Manduchi, R., Matthies, L., Owens, K., & Rankin, A. (2000, October). Terrain perception for DEMO III. Paper presented at the IEEE Intelligent Vehicles Symposium (IVS 2000), Dearborn, MI.

Biesiadecki, J., Maimone, M., & Morrison, J. (2001, June). The Athena SDM rover: a testbed for Mars rover mobility. Paper presented at the International Symposium on Artificial Intelligence (iSAIRAS 2001), St-Hubert, Canada.

Castelnovi, M., Arkin, R.C., & Collins, T.R. (2005, April). Reactive speed control system based on terrain roughness detection. Paper presented at the IEEE International Conference on Robotics and Automation (ICRA 2005), Barcelona, Spain.

Coombs, D., Lacaze, A.D., Legowik, S.A., & Murphy, K.N. (2000, October). Driving autonomously offroad up to 35 km/h. Paper presented at the IEEE Intelligent Vehicles Symposium (IVS 2000), Dearborn, MI.

Coulter, R. (1992). Implementation of the pure pursuit path tracking algorithm (Tech. Rep. CMU-RI-TR-92-01). Pittsburgh, PA: Carnegie Mellon University, Robotics Institute.

Diaz-Calderon, A., & Kelly, A. (2005). On-line stability margin and attitude estimation for dynamic articulating mobile robots. Int. J. Robot. Res., 24(10), 845–866.

Flores-Parra, I., Bienvenido, J., & Menenti, M. (2000, July). Characterization of segmentation methods by multidimensional metrics: application to the delimitation of structures. Paper presented at the IEEE National Geoscience and Remote Sensing Symposium (IGARSS 2000), Honolulu, HI.

Gillespie, T.D. (1992). Fundamentals of vehicle dynamics. Warrendale, PA, Society of Automotive Engineers.

Golda, D., Iagnemma, K., & Dubowsky, S. (2004, April). Probabilistic modeling and analysis of high-speed rough-terrain mobile robots. Paper presented at the IEEE International Conference on Robotics and Automation, New Orleans, LA.

Golberg, S., Maimone, M., & Matthies, L. (2002, March). Stereovision and rover navigation software for planetary exploration. Paper presented at the IEEE Aerospace Conference, Big Sky, MT.

Gowdy, J. (1996). IPT: An object oriented toolkit for interprocess communication (Tech. Rep. CMU-RI-TR-96-07). Pittsburgh, PA: Carnegie Mellon University, Robotics Institute.

Hart, P., Nilsson, N., & Rafael, B. (1968). A formal basis for the heuristic determination of minimum cost paths. IEEE Transactions on Systems Science and Cybernetics, 4(2), 100–107.

Harvey, W., McGlone, J., McKeown, D., & Irvine, J. (2004). User centric evaluation of automated road network extraction. Photogramm. Eng. Remote Sens., 70(12), 1353–1364.

International Standards Organization (1999). Passenger cars—test track for a severe lane-change maneuver—Part 1: Double lane-change first edition 1999-10-01. Geneva: International Standards Organization.

Jiang, T.Z., Wu, H., Wu, K., & Sun, X.W. (2005). Threshold design method of CFAR for millimeter-wave collision warning radar. Int. J. Infrared Millim. Waves, 24(3), 217–220.

Kaliyaperumal, K., Lakshmanan, S., & Kluge, K. (2001). An algorithm for detecting roads and obstacles in radar images. IEEE Transactions on Vehicle Technology, 50(1), 170–182.

Kavraki, L., Svestka, P., Latombe, J., & Overmars, M. (1996). Probabilistic roadmaps for path planning in high-dimensional configuration spaces. IEEE Trans. Rob. Autom., 12(4), 566–580.

Kelly, A., Amidi, O., Bode, M., Happold, M., Herman, H., Pilarski, T., Rander, P., Stentz, A., Vallidis, N., & Warner, R. (2004, June). Toward reliable off-road autonomous vehicle operating in challenging environments. Paper presented at the International Symposium on Experimental Robotics (ISER 2004), Singapore.

Kelly, A. & Stentz, A. (1997, January). An approach to rough terrain autonomous mobility. Paper presented at the International Conference on Mobile Planetary Robots, Santa Monica, CA.

Kelly, A. & Stentz, A. (1998). Rough terrain autonomous mobility—Part 1: A theoretical analysis of requirements. Auton. Rob., 5(2) 129–161.

Lalonde, J., Vandapel, N., & Hebert, M. (2005, June). Data structures for efficient processing in 3-D. Paper presented at Robotics: Science and Systems 1, Cambridge, MI.

LaValle, S. (1998). Rapidly exploring random trees: A new tool for path planning (Technical Report). Ames, IA: Iowa State University, Computer Science Department.

LaValle, S., & Kuffner, J. (1999, May). Randomized kinodynamic planning. Paper presented at the IEEE Inter-

national Conference on Robotics and Automation (ICRA 1999), Detroit, MI.

LaValle, S., & Kuffner, J. (2001). Rapidly exploring random trees: Progress and prospects. In B.R. Donald, K.M. Lynch, and D. Rus (Eds.), Algorithmic and computational robotics: New directions (pp. 293–308). Wellesley, MA: A. K. Peters.

Mankins, J. (1995). Technology readiness levels: A white paper. ⟨http://www.hq.nasa.gov/office/codeq/trl/trl.pdf⟩.

Nabbe, B., Kumar, S., & Hebert, M. (2004, October). Path planning with hallucinated worlds. Paper presented at IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2004), Sendai, Japan.

Nilsson, N. (1980). Principles of artificial intelligence. Palo Alto, CA: Tioga Publishing Company.

Oshkosh Truck Corp., Rockwell Collins, Rockwell Scientific, & University of Parma (2005). Team Terramax DARPA Grand Challenge 2005 (Technical Paper). Oshkosh, WI.

Roth, S.A., & Batavia, P. (2002, July). Evaluating path tracker performance for outdoor mobile robots. Paper presented at the Conference on Automation Technology for Off-Road-Equipment, Chicago, IL.

Roth, S.A., Hamner, B., Singh, S., & Hwangbo, M. (2005, July). Results in combined route traversal and collision avoidance. Paper presented at the International Conference on Field & Service Robotics (FSR '05), Port Douglas, Australia.

Shimoda, S., Kuroda, Y., & Iagnemma, K. (2005, April). Potential field navigation of high-speed unmanned ground vehicles on uneven terrain. Paper presented at the IEEE International Conference on Robotics and Automation (ICRA 2005), Barcelona, Spain.

Simmons, R., Krotkov, E., Chrisman, L., Cozman, F., Goodwin, R., Hebert, M., Katragadda, L., Koenig, S., Krishnaswamy, G., Shinoda, Y., Whittaker, W., & Klarer, P. (1995, August). Experience with rover navigation for lunarlike terrains. Paper presented at IEEE/RSJ International Conference on Robots and Systems (IROS '95), Pittsburgh, PA.

Singh, S., & Keller, P. (1991, April). Obstacle detection for high speed autonomous navigation. Paper presented at the IEEE Proceedings International Conference on Robotics and Automation (ICRA '91), Sacramento, CA.

Stanford Racing Team (2005). Stanford racing team's entry in the 2005 DARPA grand challenge (Technical Paper). Stanford, CA.

Talukder, A., Manduchi, R., Rankin, A., & Matthies, L. (2002, June). Fast and reliable obstacle detection and segmentation for cross-country navigation. Paper presented at the IEEE Intelligent Vehicle Symposium (IVS 2002). Versailles, France.

Talukder, A., Manduchi, R., Castano, R., Owens, K., Matthies, L., Castano, A., & Hogg, R. (2002, October). Autonomous terrain characterization and modeling for dynamic control of unmanned vehicles. Paper presented at the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2002). Lausanne, Switzerland.

Tompkins, P., Stentz, A., & Whittaker, W.L. (2004, March). Field experiments in mission-level path execution and replanning. Paper presented at the Conference on Intelligent Autonomous Systems (IAS-8). Amsterdam, Netherlands.

Trepagnier, P., Kinney, P., Nagel, J., Dooner, M., & Pearce, J. (2005). Team gray technical paper DARPA grand challenge 2005 (Technical Paper).

Ulrich, I., & Nourbakhsh, I. (2000, July). Appearance-based obstacle detection with monocular color vision. Paper presented at the AAAI National Conference on Artificial Intelligence. Austin, TX.

Urmson, C., Anhalt, J., Clark, M., Galatali, T., Gonzalez, J.P., Gowdy, J., Gutierrez, A., Harbaugh, S., Johnson-Roberson, M., Kato, H., Koon, P.L., Peterson, K., Smith, B.K., Spiker, S., Tryzelaar, E., & Whittaker, W.L. (2004). High-speed navigation of unrehearsed terrain: Red Team technology for Grand Challenge 2004 (Tech. Rep. CMU-RI-TR-04-37). Pittsburgh, PA: Carnegie Mellon University, Robotics Institute.

Urmson, C., Dias, M., & Simmons, R. (2002, October). Stereovision-based navigation for sun-synchronous exploration. Paper presented at the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2002), Lausanne, Switzerland.

Wettergreen, D., Cabrol, N., Baskaran, V., Calderon, F., Heys, S., Jonak, D., Luders, R.A., Pane, D., Smith, T., Teza, J., Tompkins, P., Villa, D., Williams, C., & Wagner, M.D. (2005, September). Second experiments in the robotic investigation of life in the Atacama Desert of Chile. Paper presented at the 8th International Symposium on Artificial Intelligence, Robotics and Automation in Space (iSAIRAS 2005), München, Germany.

Yamaguchi, H., Kajiwara, A., & Hayashi, S. (1998, May). Empirical study of polarization techniques for clutter reduction. Paper presented at the IEEE International Radar Conference, Dallas, TX.