System Design of an Autonomous Ground Vehicle for Participation in the 2005 DARPA Grand Challenge

Carl D. Crane III, David G. Armstrong II,
Robert Touchton, Tom Galluzzo,
Sanjay Solanki, Jaesang Lee, Daniel Kent,
Maryum Ahmed, Roberto Montane,
Shannon Ridgeway, Steve Velat,
Greg Garcia
University of Florida
Dept. of Mechanical & Aerospace Engineering
Gainesville, FL 32611
+1 352 392-9461
ccrane@ufl.edu

Michael Griffis
The Eigenpoint Company
High Springs, Florida 32655
+1 386 454-4045
griff@eigenpt.com

ABSTRACT

This paper describes the development of an autonomous vehicle system that participated in the 2005 DARPA Grand Challenge event. After a brief description of the event, the architecture, based on version 3.0 of the DoD Joint Architecture for Unmanned Systems (JAUS), and design of the system are presented in detail. In particular, the “smart sensor” concept is introduced which provided a standardized means for each sensor to present data for rapid integration and arbitration. Information about the perception sensors that were used is then presented in detail. Subsequently, testing results and performance results are presented.

Keywords

Keywords: DARPA Grand Challenge, autonomous navigation, path planning, sensor fusion, world modeling, localization, JAUS

1. INTRODUCTION

The DARPA Grand Challenge is widely recognized as the largest and most cutting-edge robotics event in the world, offering groups of highly motivated scientists and engineers across the US an opportunity to innovate in developing state-of-the-art autonomous vehicle technologies with significant military and commercial applications. The US Congress has tasked the military with making nearly one-third of all operational ground vehicles unmanned by 2015 and The DARPA Grand Challenge is one in a number of efforts to accelerate this effort. The intent of the event is to spur participation in robotics by groups of engineers and scientists outside the normal military procurement channels including leaders in collegiate research, military development, and industry research.

Team CIMAR is a collaborative effort of the University of Florida Center for Intelligent Machines and Robotics (CIMAR), The Eigenpoint Company of High Springs, Florida, and Autonomous Solutions of Young Ward, Utah. The team participated in both the 2004 and 2005 Grand Challenge events. The 2005 NaviGATOR vehicle is shown in Figure 1.

For the Grand Challenge event, teams are given a data file two hours before the start of the race that consists of the latitude and longitude of a series of waypoints that define the course. A corridor width and speed limit value are also given for each segment of the course.

2. SYSTEM ARCHITECTURE AND DESIGN

The system architecture that was implemented was based on the Joint Architecture for Unmanned Systems (JAUS) Reference Architecture, Version 3.0 [1]. JAUS defines a set of reusable components and their interfaces. The system architecture was formulated using existing JAUS-specified components wherever possible along with a JAUS-compliant inter-component messaging infrastructure. Tasks for which there are no components specified in JAUS required the creation of so-called “Experimental” components using “User-defined” messages. This approach is endorsed by the JAUS Working Group as the best way to extend and evolve the JAUS specifications.

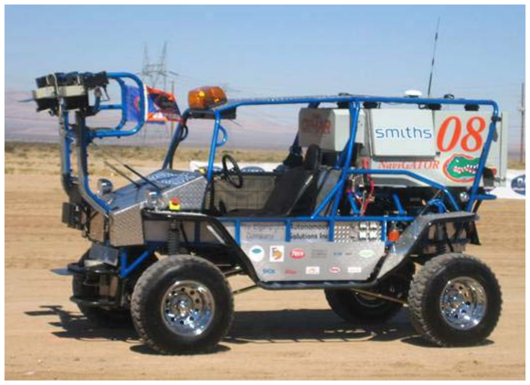
Figure 1: 2005 NaviGATOR Vehicle

2.1 High Level Architecture

At the highest level, the architecture consists of four fundamental elements:

- Planning Element: The components that act as a repository for a priori data. Known roads, trails, or obstacles, as well as acceptable vehicle workspace boundaries. Additionally, these components perform off-line planning based on that data. (represented by *world model data* and *a priori path data* boxes in Figure 2)
- Control Element: The components that perform closed-loop control in order to keep the vehicle on a specified path. (represented by *vehicle controller* box in Figure 2)
- Perception Element: The components that perform the sensing tasks required to locate obstacles and to evaluate the smoothness of terrain. (represented by *smart sensors* and *arbiter* boxes in Figure 2)
- Intelligence Element: The components that act to determine the 'best' path segment to be driven based on the sensed information. (represented by *reactive planner* box in Figure 2)

2.2 Smart Sensor Concept

The Smart Sensor concept unifies the formatting and distribution of perception data among the components that produce and/or consume it. First, a common data structure, dubbed the Traversability Grid, was devised for use by all Smart Sensors, the Smart Arbiter, and the Reactive Driver. Figure 3 shows the world as a human sees it in the upper level, while the lower level shows the Grid representation based on the fusion of sensor information. This grid was sufficiently specified to enable developers to work independently and for the Smart Arbiter to use the same approach for processing input grids no matter how many there were at any instant in time.

The basis of the Smart Sensor architecture is the idea that each sensor processes its data independently of the system and provides a logically redundant interface to the other components within the system. This allows developers to create their technologies independently of one another and process their data as best fits their system. The sensor can then be integrated into the system with minimal effort to create a robust perception system. The primary benefit of this approach is its flexibility, in effect, decoupling the development and integration efforts of the various component researchers. Its primary drawback is that it prevents the ability of one sensor component to take advantage of the results of another sensor when translating its raw input data into traversability findings.

The Traversability Grid concept is based on the well-understood notion of an Occupancy Grid, which is often attributed to Alberto Elfes of Carnegie-Mellon University [2]. His work defines an Occupancy Grid as "a probabilistic tesselated representation of spatial information." Sebastian Thrun provides an excellent treatise on how this paradigm has matured over the past 20 years as part of the Introduction to Reference 3. The expansion of the Occupancy Grid into a Traversability Grid has emerged in recent years in an attempt to expand the applicability and utility of this fundamental concept [4, 5]. The primary contribution of the Traversability Grid implementation devised for the NaviGATOR is its focus on representing degrees of traversability including terrain conditions and obstacles (from absolutely blocked to unobstructed level pavement) while preserving real-time performance of 20 Hz.

The Traversability Grid design is 121 rows (0 – 120) by 121 columns (0 – 120), with each grid cell representing a half-meter by half-meter area. The vehicle occupies the center cell at location (60, 60). The sensor results are oriented in a global frame of reference so that North is always the top of the grid. In this fashion, a 60m by 60m grid is produced that is able to accept data at least 30m ahead of the vehicle and store data at least 30m behind it. To support proper treatment of the vehicle's position and orientation, every Smart Sensor component is responsible for establishing a near-real-time latitude/longitude and heading (yaw) feed from the GPOS (global positioning)component.

The scoring of each cell is based on mapping the sensor's assessment of the traversability of that cell into a range of 2 to 12 where 2 means that there is absolutely an insurmountable obstacle detected in that cell, 12 means there is absolutely a desirable,

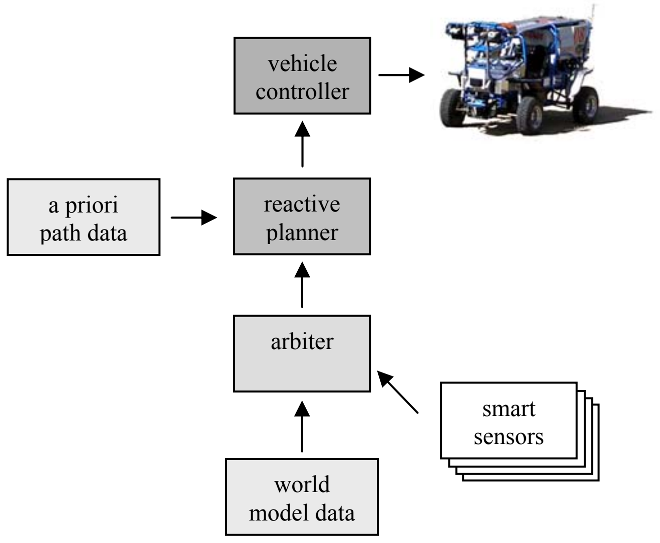

Figure 2: High Level Architecture

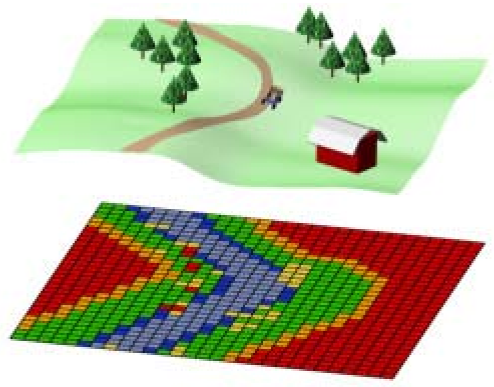

Figure 3: Traversability Grid

easily traversed surface in that cell, and 7 means that the sensor has no evidence that the traversability of that cell is particularly good or bad. Certain other values are reserved for use as follows: 0 → "out-of-bounds," 1 → "use what was sent last time," 13 → "failed/error," 14 → "unknown," and 15 → "vehicle location." These discrete values have been color-coded to help humans visualize the contents of a given Traversability Grid, from red (2) to gray (7) to green (12).

All of these characteristics are the same for grids sent from every Smart Sensor, making seamless integration possible, with no predetermined number of sensors. The grids are sent to the Smart Arbiter, which is responsible for fusing the data. The arbiter then sends a grid with all the same characteristics to the Reactive Driver, which uses it to dynamically compute the desired vehicle speed and heading.

The messaging concept for marshalling grid cell data from sensors to the arbiter and from the arbiter to the reactive driver is to send an entire traversability grid as often as the downstream component has requested it (typically at 20 Hz). In order to properly align a given sensor's output with that of the other sensors, the message must also provide the latitude and longitude of the center cell (i.e., vehicle position at the instant the message and its cell values were determined).

In order to aid in the understanding, tuning, and validation of the Traversability Grids being produced, a Smart Sensor Visualizer (SSV) component was developed. Used primarily for testing, the SSV can be pointed at any of the Smart Sensors, the Smart Arbiter, or the Reactive Driver and it will display the color-coded Traversability Grid, along with the associated vehicle position, heading, and speed. The refresh rate of the images is adjustable from real-time (e.g., 20 Hz) down to periodic snapshots (e.g., 1 second interval).

2.3 Concept of Operation

The most daunting task of all was integrating the components such that an overall mission could be accomplished. Figure 3 portrays schematically how the key components work together to control the vehicle. Figure 4 also shows how the Traversability Grid concept enables the various Smart Sensors to deliver grids to the Smart Arbiter, which fuses them and delivers a single grid to the Reactive Driver. Prior to beginning a given mission, the a priori Planner builds the initial path, which it stores in a Path File as a series of GPS waypoints. Once the mission is begun, the Reactive Driver sequentially guides the vehicle to each waypoint in the Path File via the Primitive Driver. Meanwhile, the various Smart Sensors begin their search for obstacles and/or smooth surfaces and feed their findings to the Smart Arbiter. The Smart Arbiter performs its data fusion task and sends the results to the Reactive Driver. The Reactive Driver looks for interferences or opportunities based on the feed from the Smart Arbiter and alters its command to the Primitive Driver accordingly. Finally, the goal is to perform this sequence iteratively on a sub-second cycle time (10 to 60 Hz), depending on the component, with 20 Hz as the default operational rate.

3. PERCEPTION

This section of the paper discusses how the NaviGATOR collects, processes and combines sensor data. Each of the sensor components is presented, organized by type: LADAR, camera, or

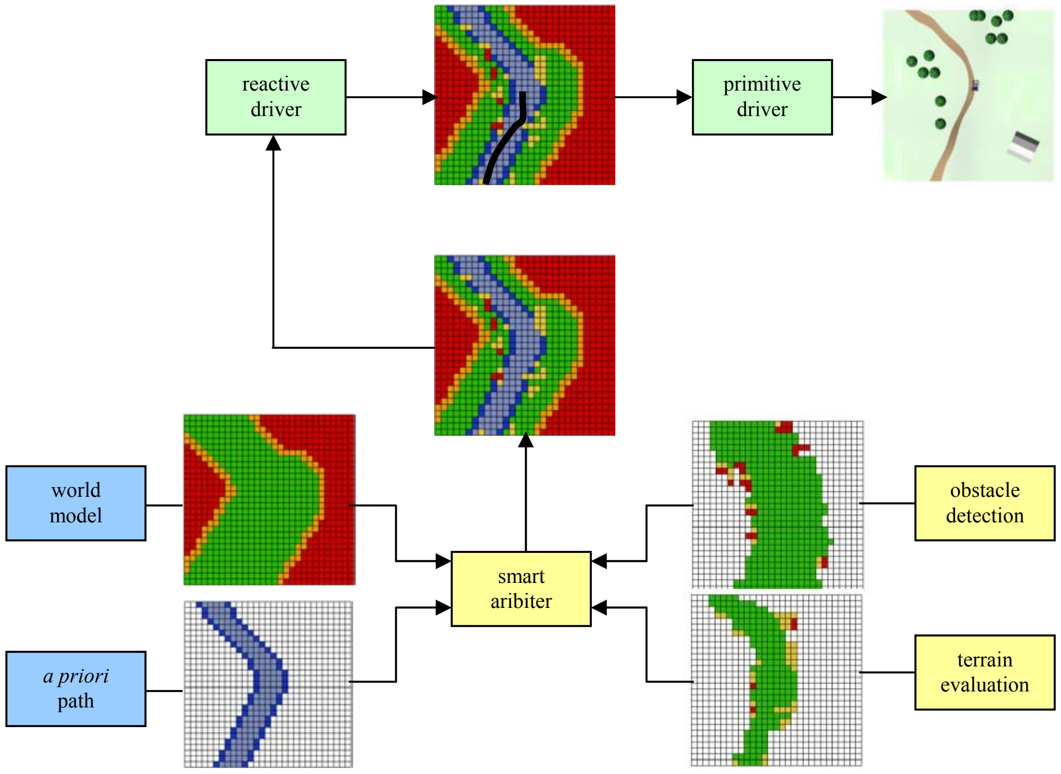

Figure 4: Operational Schematic (including Traversabiliy Grid Propagation)

“pseudo” (a component that produces an output as if it were a sensor, but based on data from a file or database). Finally, the Smart Arbiter sensor fusion component is discussed.

3.1 LADAR-based Smart Sensors

There are three Smart Sensors that rely on LADAR range data to produce their results: the Terrain Smart Sensor (TSS), the Negative Obstacle Smart Sensor (NOSS) and the Planar LADAR Smart Sensor (PLSS) (see Figure 5). All three components use the LMS291-S05 from Sick Inc. for range measurement. The TSS will be described in detail and then the remaining two will be discussed only in terms of how they are different than the TSS.

A laser range finder operates on the principle of time of flight. The sensor emits an eye-safe infrared laser beam in a single-line sweep of either 180° or 100°, detects the returns at each point of resolution, and then computes single line range image. Although three range resolutions are possible (1°, 0.5° or 0.25°) the resolution of 0.25° can only be achieved with a 100° range scan. The accuracy of the laser measurement is +/- 50 mm for a range of 1 to 20 m. A high-speed serial interface card is used to achieve the needed high-speed baud rate of 500 kB.

3.1.1 Terrain Smart Sensor (TSS)

The sensor is mounted facing forward at an angle of 6° towards the ground. For the implementation of the TSS, the 100° range with a 0.25° resolution is used. With this configuration and for nominal conditions (flat ground surface, vehicle level), the laser scans at a distance of ~20 m ahead of the vehicle and ~32 m wide. The TSS converts the range data reported by the laser in

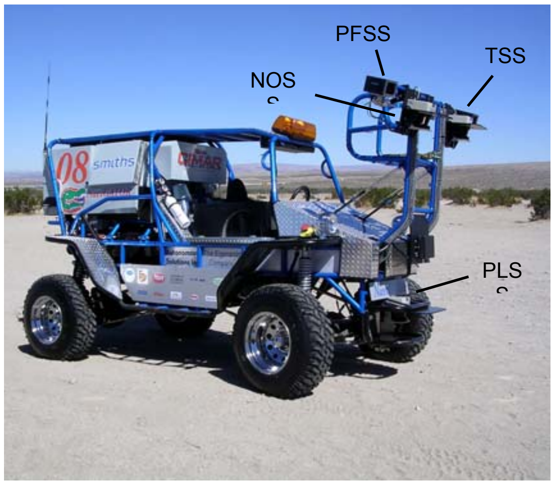

Figure 5: Perception Sensors

polar coordinates into Cartesian coordinates local to the sensor, with the Z-axis vertically downward and the X-axis in the direction of vehicle travel. The height for each data point (Z-component) is computed based on the known geometry of the system and the range distance being reported by the sensor. The data is then transformed into the global coordinate system required by the Traversability Grid, where the origin is the centerline of the vehicle at ground level below the rear axle (i.e., the projection of the GPS antenna onto the ground), based on the instantaneous roll, pitch, and yaw of the vehicle.

Each cell in the Traversability Grid is evaluated individually and classified for its traversability value. The criteria used for classification are:

1. The mean elevation (height) of the data point(s) within the cell.
2. The slope of the best fitting plane through the data points in each cell.
3. The variance of the elevation of the data points within the cell

A traversability value between 2 and 12 is assigned to each cell, depending on the severity values of the mean height, slope, and variance information. A cell must contain a minimum of three data points or else that cell is flagged as unknown. This also helps in eliminating noise. Each of the parameters is individually mapped to a corresponding traversability value for a given cell. This mapping is entirely empirical and non-linear. A weighted average of these three resulting traversability values is used to assign the final traversability value.

3.1.2 Negative Obstacle Smart Sensor (NOSS)

The NOSS was specifically implemented to detect negative obstacles (although it can also provide information on positive obstacles and surface smoothness like the TSS). The sensor is configured like the TSS, but at an angle of 12° towards the ground. With this configuration and for nominal conditions, the laser scans the ground at a distance of ~10 m ahead of the vehicle. To detect a negative obstacle, the component analyzes the cases where it receives a range value greater than would be expected for level ground. In such cases the cell where one would expect to receive a hit is found by assuming a perfectly horizontal imaginary plane. This cell is found by solving for the intersection of the imaginary horizontal plane and the line formed by the laser beam. A traversability value is assigned to that cell based on the value of the range distance and other configurable parameters. Thus a negative obstacle is reported for any cell whose associated range data is greater than that expected for an assumed horizontal surface. The remaining cells for which range value data is received are evaluated on a basis similar to the TSS.

3.1.3 Planar LADAR Smart Sensor (PLSS)

The sensor is mounted 0.6 m above the ground, scanning in a plane horizontal to the ground. Accordingly, the PLSS only identifies positive obstacles and renders no opinion regarding the smoothness or traversability of areas where no positive obstacle is reported. For the PLSS, the 180° range with a 0.5° resolution is used. The range data from the laser is converted into the Global coordinate system and the cell from which each hit is received is identified. Accordingly the “number of hits” in that cell is incremented by one and then, for all the cells between the hit cell and the sensor, the “number of missed hits” is incremented by one. Bresenham’s line algorithm is used to efficiently determine the indices of the intervening cells.

A traversability value between 2 and 7 is assigned to each cell based on the total number of hits and misses accumulated for that cell. The mapping algorithm first computes a score, which is the difference between the total number of hits and a discounted number of misses in a cell (a discount weight of 1/6 was used for the event). This score is then mapped to a traversability value using an exponential scale of 2. For example, a score of 2 or below is mapped to a traversability value of "7," a score of 4 and below is mapped to a "6" and so on, with a score greater than 32 mapped to a "2." The discounting of missed hits provides conservatism in identifying obstacles, but does allow gradual recovery from false positives (e.g., ground effects) and moving obstacles.

3.2 Camera-based Smart Sensor

The Pathfinder Smart Sensor (PFSS) consists of a single color camera mounted in the sensor cage and aimed at the terrain in front of the vehicle. Its purpose is to assess the area in the camera's scene for terrain which is similar to that on which the vehicle is currently traveling, and then translate that scene information into traversability information. The PFSS component uses a high-speed frame-grabber to store camera images at 30 Hz.

Note that the primary feature used for analytical processing is the RGB (Red, Green, and Blue) color space. This is the standard representation in the world of computers and digital cameras and is therefore often a natural choice for color representation. Also RGB is the standard output from a CCD-camera. Since roads typically have a different color than non-drivable terrain, color is a highly relevant feature for segmentation. The following paragraphs describe the scene assessment procedure applied to each image for rendering the Traversability Grid that is sent to the Smart Arbiter.

To reduce the computational expense of processing large images, the dimensions of the scene are reduced from the original digital input of 720 × 480 pixels to a 320 × 240 reduced image. Then, the image is further preprocessed to eliminate the portion of the scene that most likely corresponds to the sky. The segmentation of the image is based simply on physical location within the scene (tuned based on field testing), adjusted by the instantaneous vehicle pitch. This very simplistic approach is viable because the consequences of inadvertently eliminating ground are minimal due to the fact that ground areas near the horizon will likely be beyond the 30 m planning distance of the system. The motivation for this step in the procedure is that the sky portion of the image hinders the classification procedure in two ways. First, considering the sky portion slows down the image processing speed by spending resources evaluating pixels that could never be drivable by a ground vehicle. Second, there could be situations where parts of the sky image could be misclassified as road.

Next, a 100 × 80 sub-image is used to define the drivable area and two 35 × 50 sub-images are used to define the background. The drivable sub-image is placed in the bottom, center of the image while the background sub-images are placed at the middle-right and middle-left of the image, which is normally where the background area will be found, based on experience [7] (see Figure 6). When the vehicle turns, the background area that is in the direction of the turn will be reclassified as a drivable area. In this case, that background area information is treated as road area by the classification algorithm.

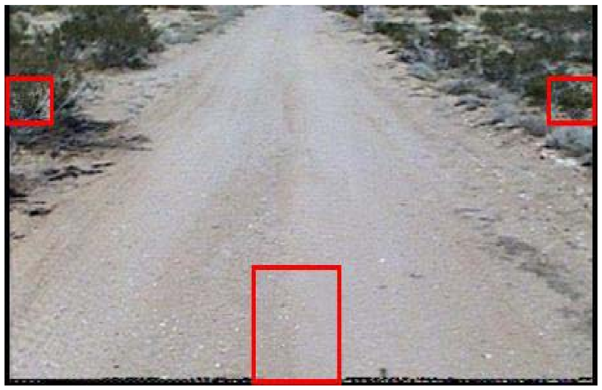
Figure 6: Scene Segmentation Scheme

A Bayesian decision theory approach was selected for use, as this is a fundamental statistical approach to the problem of pattern classification associated with applications such as this. It makes the assumption that the decision problem is posed in probabilistic terms, and that all of the relevant probability values are known. The basic idea underlying Bayesian decision theory is very simple. However this is the optimal decision theory under Gaussian distribution assumption [8].

A block-based segmentation method is used to reduce the segmentation processing time. 4×4 pixel regions are clustered together and replaced by their RGB mean value. The clusters, or blocks, are then segmented, and the result, as shown in Figure 7 (a), has less noise compared with pixel based approaches, Figure 7 (b). Also the segmentation process is accomplished faster than pixel-based classification. A disadvantage, however, is that edges are jagged and not as distinct.

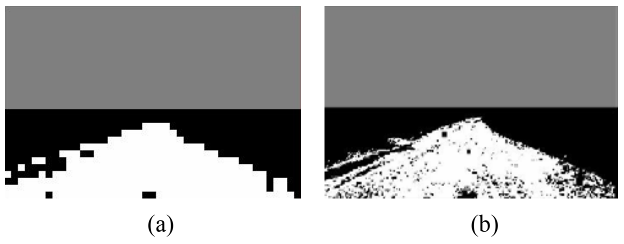

Figure 7: Processed Images

After processing the image, the areas classified as drivable road are converted by perspective transformation estimation into the global coordinates used for the Traversability Grid [9]. The perspective transformation matrix is calculated based on camera calibration parameters and the instantaneous vehicle heading. Finally, the PFSS assigns a value of 12 (highly traversability) to those cells that correspond to an area that has been classified as drivable. All other cells are given a value of 7 (neutral). Figure 8 depicts the PFSS Traversability Grid data after transformation into global coordinates.

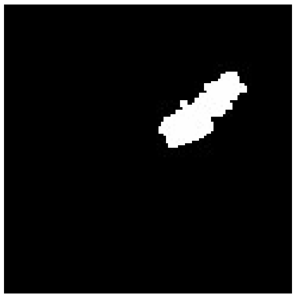

Figure 8: Transformed Image

3.3 Pseudo Smart Sensors

There are two Smart Sensors that produce Traversability Grids based on stored data: the Boundary Smart Sensor (BSS) and the Path Smart Sensor (PSS).

The BSS translates boundary knowledge, defined as boundary polygons prior to mission start, into real-time Traversability Grids, which assures that the vehicle does not travel outside the given bounds. The BSS is responsible for obtaining the boundary information from a local spatial database. The BSS uses this data to determine the in-bounds and out-of-bounds portions of the traversability grid for the instantaneous location of the vehicle. The BSS also has a configurable "feathering" capability that allows the edge of the boundary to be softened, creating a buffer area along the edges. This feature provides resilience to uncertainties in the position data reported by the GPOS component. Figure 9 shows a typical grid output from the BSS indicating the vehicle's location within the grid and the drivable region around it. By clearly demarking areas of the grid as out-of-bounds, the BSS allows the Smart Arbiter to summarily dismiss computation of out-of-bounds grid cells and the Reactive Driver to prune its search tree of potential plans.

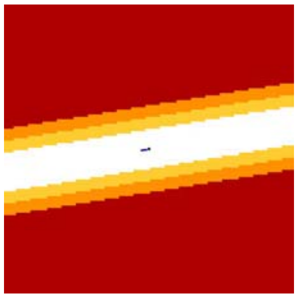

Figure 9: Traversability Grid Showing Boundary Data

The PSS translates the *a priori* path plan, stored as a "path file" prior to mission start, into real-time Traversability Grids. The PSS uses this path data to superimpose the originally planned path onto the traversability grid based on the instantaneous location of the vehicle. The PSS has a configurable "feathering" capability that allows the width of the path to be adjusted and the edges of the path to be softened. This feature also allows the engineer to select how strongly the originally planned path should be weighted by setting the grid value for the centerline. A 12 would cause the arbiter and planner to lean towards following the original plan even if the sensors were detecting a better path, while a 10, which is what was used at run-time, would make the original plan more like a suggestion that could be more easily overridden by sensor findings. Figure 10 shows a typical grid output from the PSS indicating the vehicle's location within the grid and the feathered *a priori* planned path flowing through the in-bounds corridor.

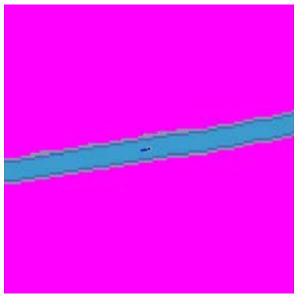

Figure 10: Traversability Grid Showing *a Priori* Path Data

3.4 Sensor Fusion

With the Traversability Grid concept in place to normalize the outputs of a wide variety of sensors, the data fusion task becomes one of arbitrating the matching cells into a single output finding for that cell for every in-bounds cell location in the grid. To accomplish this, the Smart Arbiter must receive and unpack each new message from a given sensor and then adjust its center-point to match that of the Arbiter (assuming that the vehicle has moved between the instant in time when the sensor's message was built and the instant in time when the arbitrated output message is being built). This step must be repeated for each sensor that has sent a message. At this point, all input grids are aligned and contain the latest findings from its source sensor. Now the arbiter must simultaneously traverse the input grids, cell-by-cell, and merge the data from each corresponding cell into a single output value for that row/column location. Once all cells have been treated in this fashion, the Smart Arbiter packs up its output grid message and sends it on the Reactive Driver.

For early testing, a simple average of the input cell values was used as the output cell value. Later work investigated other algorithms, including heuristic ones, to perform the data fusion task. The Arbiter component was designed to make it easy to experiment with varying fusion algorithms in support of on-going research. The algorithm that was used for the DGC event used a two-stage heuristic approach. Stage 1 is an "auction" for definite obstacles to see if any sensor reported a "2" and, barring that, did any sensor report a "3" for the cell position under consideration.

Stage 2 then depends on the results of the "auction." If no sensor "wins" the auction, then all of the input cells at that position are averaged, including the arbiter's previous output value for that cell. If any sensor reported a "2" in Stage 1, then all other sensors are ignored and the arbiter's previous output value for that cell is decremented by a configurable amount (which was 2 at run-time). For an auction winner of "3", the decrement size is reduced by half (or just 1 at run-time). Thus, a sensor must report a "2" for several iterations in order for the arbiter to lower its output value to a "2", thus providing a dampening effect to help circumvent thrashing in a sensor's output values. The averaging of input values along with the arbiter's previous output value also provides a dampening effect.

The premise used for all algorithms that were explored was to keep the arithmetic very simple and in-place since the data fusion task demands can reach 2 million operations per second just to process the algorithm. Thus, complex, probabilistic-based and belief-based approaches were not explored. However, adding highly traversable cells to the auction (i.e., 11' and 12's) and post-processing the output gird to provide proximity smoothing and/or obstacle dilation were explored, but none of these alternatives provided any better performance (in the sense of speed or accuracy) than the one used for the event.

4. REAL-TIME PLANNING AND VEHICLE CONTROL

The purpose of online planning and control is to autonomously drive the NaviGATOR through its sensed environment along a path that will yield the greatest chance of successful traversal. This functionality is compartmentalized into the Reactive Driver (RD) component of the NaviGATOR. The data input to this component include the sensed cumulative traversability grid, assembled by the Smart Arbiter component, vehicle state information, such as position and velocity, and finally the *a priori* path plan, which expresses the desired path for the vehicle to follow *sans* sensor input. Given this information, the online real-time planning and control component, seeks to generate low-level actuator commands, which will guide the vehicle along the best available path, while avoiding any areas sensed as poorly traversable.

The controller attempts to optimize the cost of the trajectory by employing an iterative deepening A* search algorithm. The goal of the search is to find a set of open-loop actuator commands that minimize the cost of the trajectory through the traversability space, and also bring the vehicle to within a given proximity of a desired goal state. The goal state is estimated as the intersection of the *a priori* path with the boundary of the traversability grid. As the vehicle nears the end of the path, and there is no longer an intersection with the grid boundary, the desired goal state is simply the endpoint of the last *a priori* path segment.

Closed loop control with the receding horizon controller is achieved by repeating the optimization algorithm as new traversability data are sensed and vehicle state information is updated. Thus disturbances, such as unanticipated changes in traversability or vehicle state, are rejected by continually reproducing a set of near optimal open loop commands at 20 Hz, or higher.

The search calculates different trajectories by generating input commands and extrapolating them through a vehicle kinematics model. The cost of the resulting trajectory is then calculated by integrating the transformed traversability value along the geometric path that is produced through the grid. The search continues until a solution set of open-loop commands is found that produce a near optimal trajectory. The first command in the set is then sent to the actuators, and the process is repeated. A typical result of the planning optimization is shown in (see Figure 11, where the dark line is the final instantaneous solution).

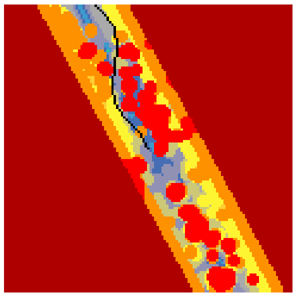

Figure 11: Sample Planning Result through Traversability Grid

5. TESTING AND PERFORMANCE

5.1 Testing in Preparation for the Grand Challenge

Testing began with the JAUS messaging system on the ten computers that would drive the NaviGATOR. The JAUS messaging would need to be capable of sending up to 500 messages per second per node for over 14 hours. On race day, over 20 million JAUS messages were actually sent and received. Initial testing of the individual JAUS components, discussed in this paper, took place in the spring of 2005 primarily in the CIMAR lab at the University of Florida. The goal was to get each component working by itself, "on the bench" in a controlled laboratory environment.

The first field testing was conducted at the University of Florida's Plant Science Unit located in Citra, Florida. A course was laid out in an open field and consisted mainly of a figure eight, an oval, and several left and right sharp turns. Various segments were added to this course to replicate terrain that was expected in the desert.

On 11 September 2005, the NaviGATOR was transported west to the Stoddard Valley Off Highway Vehicle (OHV) Area near Barstow, California. The Stoddard Valley OHV consists of hundreds of miles of graded roads and dirt trails. These roads and trails are an order of magnitude more difficult to navigate than the test track at Citra and less forgiving of mistakes. They have large ruts and washouts and in some places, they wind through the mountains with a hill on one side of the road and a steep cliff on the other. The best aspect of testing out in the desert was getting out of the oval and stretching out on test routes that were up to 20 miles long. After two weeks of rigorous, dawn-to-dusk testing in Stoddard Valley, the NaviGATOR had gone from a personal best of 12 miles at 10mph on a track in Florida to running autonomously over 40 miles of rough desert terrain at speeds reaching 30 mph.

5.2 The National Qualification Event

The qualification course at the California Speedway is shown in Figure 12. It consisted of a 2.3 mile long path with three parked cars, a rough terrain section, a simulated mountain pass, a tunnel, and finally a wooden "tank trap" obstacle.

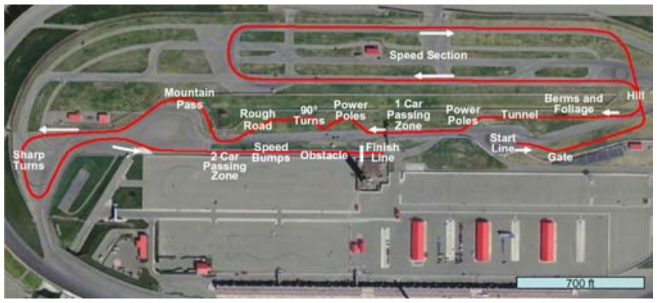

Figure 12: Qualification Course at California Speedway

The NaviGATOR completed the entire course on the first attempt. However, three lane-marking cones had been hit and the tank trap obstacle at the end of the course had been slightly brushed. Two changes were made to the NaviGATOR for the second run. The desired speed on the high-speed section of the course was increased from 16 mph to 20 mph and the dilated size of the perceived obstacles was increased in an attempt to completely miss the tank trap obstacle. During the second run, the NaviGATOR began oscillating and became unstable on the high-speed section and the run was aborted. The problem was that the high-speed section of the qualification course was on pavement whereas all high-speed testing had been conducted off-road. The disturbances caused by the constant four-wheel drive on pavement were responsible for the oscillation.

For the third run on the qualification course, all parameters were reset to those used during the first run. All went well until the vehicle scraped the concrete wall in the mountain pass section of the course, snapping the front steering linkage. The vehicle was quickly repaired. For future runs, the path centerline as reported by the Path Smart Sensor (PSS) was shifted twelve inches away from the wall in the mountain pass section. After this, the qualification course was successfully completed two more times. In summary, the NaviGATOR completed the entire qualification course three out of five times and the team was selected by DARPA to compete in the desert race.

5.3 The DARPA Grand Challenge Race

The team received the course data file containing the course waypoints in the early morning of 8 October 2005. Two hours were allocated for processing the data, which primarily consisted of setting desired speeds for each section of the course. The path file was then uploaded to the vehicle and by 9:30 am the NaviGATOR was off. After leaving the start gate, the NaviGATOR headed off into the desert and then circled around past the crowd at about the eight-mile mark. The NaviGATOR headed past the spectators at approximately 24 mph, performing very well at this point in the race (see Figure 13).

The NaviGATOR next flawlessly traversed a bridge over a railroad track and disappeared into the brown desert haze. Shortly before 11 a.m., the team received word from the chase truck that was following NaviGATOR that the vehicle had inexplicably run off the road and stopped. NaviGATOR appeared reluctant to move forward into and out of low brush in front of it, although its off-road capabilities would have easily carried it through. After several attempts to pause and restart the NaviGATOR, the driver called back to say the vehicle was moving, but slowly and still off the road. After about a half mile of starting, stopping, and driving

Figure 13: NaviGATOR Passing the Stands at the 2005 DARPA Grand Challenge Event

very slowly over brush, it regained the road and took off again at high speed following the road perfectly. However, after about another mile, the vehicle again went off the road and this time stopped in front of a bush. This time, DARPA officials quickly declared the NaviGATOR dead. The time was shortly before noon, and NaviGATOR had traveled past the 24-mile marker. NaviGATOR placed 18th among the 23 finalists. A total of five teams actually completed the entire course, with Stanford's Stanley taking the $2 million prize for the shortest time of six hours, 53 minutes and 58 seconds.

5.4 What stopped the NaviGATOR?

Team members went out on the course the day following the race and found the NaviGATOR tire tracks at the two locations where the vehicle went off the right side of the road. From this information and data that was logged on the vehicle, it appears that the calculated GPS position drifted by approximately twenty feet causing the vehicle to want to move to the right of the actual road. From the tire tracks and from the traversability grid (see Figures 14 and 15), it was apparent that the vehicle wanted to move to the right, but the obstacle avoidance sensors were detecting the bushes and berms on the right side of the road. From the vehicle's perspective (see Figure 14) it appeared that the

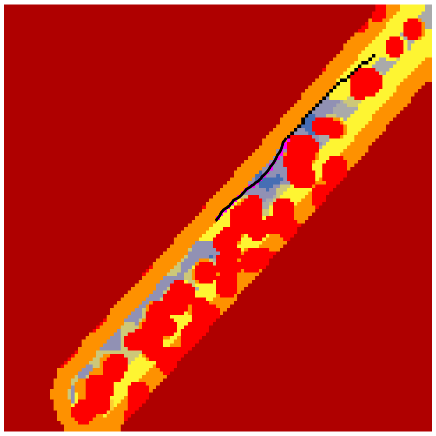

Figure 14: Traversability Grid
(during time of position system drift)

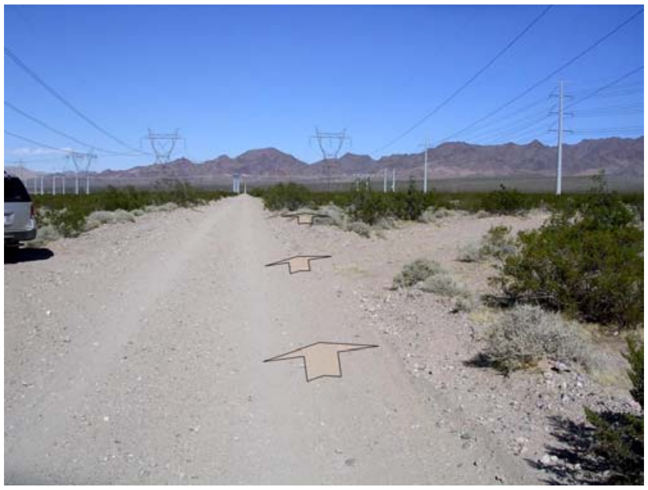

Figure 15: Location where NaviGATOR veered off the course and was stopped.

corridor was littered with objects and the best it could do was to travel along the left side of the corridor on the verge of going out of bounds on the left. In reality, the vehicle was hugging the right side of a very navigable dirt road, however most of the open road was being classified as out of bounds.

Both times that the vehicle went off course were due to the fact that the right side became free of obstacles and the vehicle attempted to move to the center of it*s incorrect corridor*. Figure 15 shows the location where the NaviGATOR moved off the course for the second time whereupon DARPA officials stopped it.

6. CONCLUSIONS AND FUTURE WORK

Overall the performance of the NaviGATOR system has been very good. The base vehicle is very capable and has excellent mobility in very rough terrain. The obstacle and terrain detection sensors and sensor integration approach worked very well as did the reactive planner module. Overall, the control loop (from sensed objects to determination of vehicle actuation parameters) operated at a rate of over 20 Hz. Also, a significant contribution of the effort was to show that JAUS could be used successfully in a situation such as this and that the standardized messaging system defined by JAUS could greatly simplify the overall integration effort.

In retrospect, the team would have benefited from more testing time in the California desert. The issues associated with the positioning system and the high-speed control on pavement could have been resolved. However, the project was very successful in that an entirely new vehicle system was designed, fabricated, and automated in a nine-month period, ready to compete in the 2005 DARPA Grand Challenge. This was a monumental effort put on an aggressive time and resource schedule.

7. ACKNOWLEDGEMENTS

The authors would like to acknowledge the work of all the DARPA staff members who organized and operated a very successful event. Also thanks are given to the members of the JAUS Working Group whose efforts are leading to an open architecture that will promote interoperability of military robotic systems. Finally, the authors gratefully acknowledge the technical and financial support of the team's Title Sponsor, Smiths Aerospace.

8. REFERENCES

[1] Joint Architecture for Unmanned Systems (JAUS) Reference Architecture, Version 3.0, http://www.jauswg.org/.

[2] Elfes, A., "Using Occupancy Grids for Mobile Robot Perception and Navigation," Computer Magazine, IEEE, pp 46-57, June 1989.

[3] Thrun, S., "Learning Occupancy Grid Maps with Forward Sensor Models," Autonomous Robots 15, 111-127, 2003.

[4] Seraji, H., "New Traversability Indices and Traversability Grid for Integrated Sensor/Map-Based Navigation," Journal of Robotic Systems 20(3), 121-134, 2003.

[5] Ye, C., and Borenstein, J., "T-transformation: Traversability Analysis for Navigation on Rugged Terrain," Proceedings of the Defense and Security Symposium, Unmanned Ground Vehicle Technology VI (OR54), Orlando Fl, 2004.

[6] Solanki, S., "Implementation of laser range sensor and dielectric laser mirrors for 3D scanning of glove box environment," Master's Thesis, University of Florida, 2003.

[7] Lee, J., Crane, C., Kim, S., and Kim J., "Road Following in an Unstructured Desert Environment using Monocular Color Vision as Applied to the DARPA Grand Challenge," proceedings of ICCAS2005, Seoul, 2004.

[8] Morris, R.D., Descombes, X., and Zerubia, J., "Fully Bayesian image segmentation - an engineering perspective" In Proceedings of IEEE International Conference on Image Processing, Santa Barbara, October 1997.

[9] http://www.robots.ox.ac.uk/~vgg/presentations/bmvc97/criminispaper/ , "A Plane Measuring Device", A. Criminisi, I. Reid, and A. Zisserman at the University of Oxford.