

Draft v1 1/9/06

Prospect Eleven

Princeton University's Entry in the 2005 DARPA Grand Challenge

by
Alain L. Kornhauser¹, Brendan Collins², Gordon Franken², Andrew Saxe², Anand
Atraye³, Bryan Cattle³, and Scott Schiffres⁴

1. Background

Prospect Eleven is the vehicle name for Princeton University's entry in [DARPA's Grand Challenge of 2005](#), a competition of truly autonomous vehicles traveling a prescribed course. The challenge of the "Challenge" was to "build or modify" an automobile-sized vehicle that can negotiate a prescribed course containing randomly placed obstacles without any human intervention. The Challenge was originally contested in March, 2004; however, none of the entries completed the course. In fact, the most successful vehicle completed only the first seven miles of the more than 140 mile course. As a result, [DARPA](#) (Defense Advanced Research Projects Administration), decided to organize a second Challenge which took place on October 8, 2005.

Princeton University did not participate in the first Challenge; however, upon learning, in April 2004, that a second Challenge would be contested, a group of undergraduates led by Ben Klaber'05 approached Professor Alain Kornhauser with a desire to participate in the 2005 Challenge. In May 2004, Princeton University officially entered a vehicle named "*Prospect Eleven*" with the stipulations that it be an undergraduate student, as opposed to a professional staff, activity **and** that the principle objective be that it complement to the highest degree possible the students' academic experience at Princeton. Throughout the design, build and test process leading to the competition and beyond, the guiding objectives have been academic relevance, simplicity, elegance and minimal expenditure of funds. Academic relevance because it is Princeton; simplicity because it is too easy to make this initiative so hard that it is undoable; thus, the need for elegance. Finally, funds were to be used only for summer stipends for participating students, the purchase of needed computing, command and control apparatus and travel expenses associated with the competition and post-competition activities. No funds were available for students during the academic year, professional staff, nor advising faculty. Student participation during the academic year had to be justifiable for academic credit or as an extra-curricular activity. In total, approximately \$125,000 was spent over a period of eighteen months. 40 % was for summer stipends for students during the summers of 2004 and 2005. \$25,000 was spent on travel expenses associated with the Challenge and a return

¹ Team leader, Professor Operations Research & Financial Engineering, Princeton University

² Class 2008, Princeton University

³ Class 2007, Princeton University

⁴ Class 2006, Princeton University

to the desert three weeks later. The rest paid for associated equipment including the stereo cameras, computers, all digital-mechanical control elements and miscellaneous expenses. The source of funds came from the CSX Transportation Research Endowment Fund and substantial cash gifts from Eric Huber and Katherine Kornhauser P03. Finally, the base vehicle was donated by General Motors, Applanix Corporation loaned a GPS inertial navigation unit, Tom Haines Pick-your-own Berry Farm provided one of the test facilities and transport of Prospect Eleven to and from the desert in an enclosed van was paid by General Motors and ALK Technologies, Inc. What follows is a high-level description of some of the main systems of Prospect Eleven and a description of its accomplishments.

2. Digital-Mechanical Systems.

In the hierarchical setup of an autonomous vehicle system, the lowest level consists of the fundamental drive-by-wire modifications that enable the mechanical systems on the vehicle to be controlled by digital commands. Making a vehicle drive-by-wire requires a comprehensive knowledge of the workings of the vehicle itself, as well as a strong understanding of how the components will interface with the rest of the autonomous system.



Figure 2.1 Prospect Eleven's Stereo camera and GPS boom

A stock 2005 GMC Canyon was used as the vehicle platform. Salvaged by General Motors (GM) due to it being damaged in transit to a dealership, it was donated to Princeton University. Dubbed "Prospect Eleven", this silver Crew Cab 4-wheel drive pickup truck was transformed by the students into a powerful and reliable autonomous vehicle. The most challenging constraint on the transformation was the self-imposed mandate that the vehicle remain completely human drivable and street legal.

The digital-mechanical systems installed into an autonomous vehicle have the overall purpose of duplicating their corresponding human-mechanical components. These systems essentially put feet on the pedals and hands on the wheel. Additionally, the transmission needed shifting, and a fail-safe emergency brake was needed to ensure the safety of the vehicle and those around it, should any mishaps occur. The human-mechanical systems that were extended to also become digital-mechanical systems were:

- Accelerator Pedal
- Brake Pedal
- Transmission
- Emergency Brake
- Steering Wheel

Unlike many teams in the 2005 DARPA Grand Challenge, the Princeton team received essentially no help from industry or corporate sponsors. Each of the electro-mechanical systems on Prospect Eleven, was conceived, designed, fabricated & tested by the team of undergraduates. Cost-effective, simple, custom designs were essential implementation objectives.

2.1 Accelerator Pedal. As a new vehicle, Prospect Eleven had many of the advanced features of other commercially available cars, including advanced electronics systems. As such, the engine had already been redesigned to be more electronic and more efficient. The throttle valve is entirely electronic and has no mechanical link to the accelerator pedal. The pedal itself is simply a voltage divider, acting as an analog input into the truck’s engine control computer. While the pedal could be activated mechanically, through an actuator or linkage, it was also possible and much simpler to use a computer-generated voltage to simulate the sensor response of the pedal. Human drivability was maintained through the use of a simple A/B switch; switching the signal lines between Prospect Eleven’s computers and the actual pedal.

The electronic interface was not without its challenges. Much difficulty was encountered when it came to calibrating the simulated voltage to behave like the pedal generated signal. This became a substantial “reverse engineering” project to correlate the analog signals with accelerator pedal position.

2.2 Brake Pedal. To complement the accelerator, and provide complete speed control for Prospect Eleven, it was necessary to have the ability to actuate the brake pedal. Since the brakes (and steering) system(s) were conventional no opportunity existed to simply brake (or steer) using only an electronic interface. What needed to be designed and built was an electro-mechanical device that could be activated electronically to provide the mechanical application of the brakes (and steering). Early on, it was decided to leave all of the truck’s brake-related systems intact, and simply actuate the pedal itself. The need to leave the vehicle human-drivable eliminated many actuator designs that required a permanent rigid connection to the pedal. Instead, actuation was accomplished by simply pulling on the brake using a cable. Restoration was provided by a spring. This allows a human to depress the pedal at will; instead of fighting against an actuator, the cable simply goes slack as the pedal is pressed. See Figure 2.2

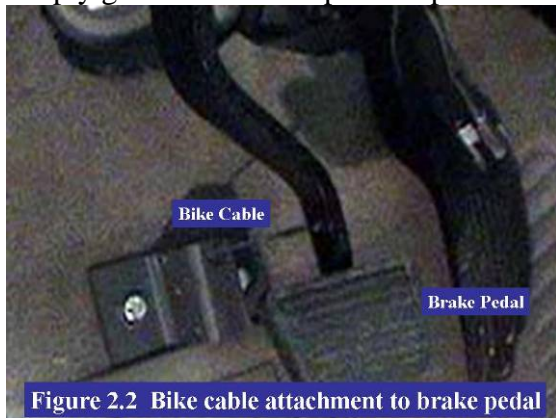


Figure 2.2 Bike cable attachment to brake pedal

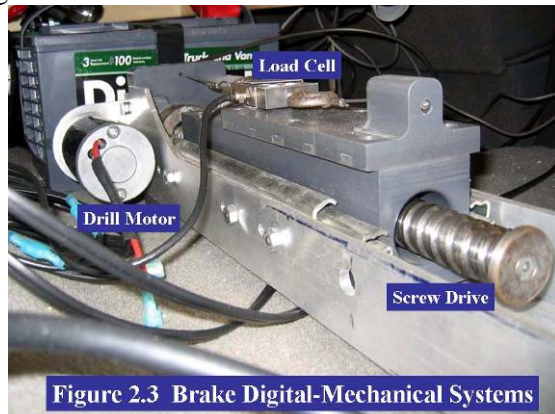


Figure 2.3 Brake Digital-Mechanical Systems

The cable is controlled by a custom-built linear ball-screw actuator, driven by a 12 Volt drill motor, see Figure 2.3. This actuator is mounted on the floor of the cab, behind the driver's seat; this positioning allows the actuator to remain low-profile, yet accessible. However, it also requires that the cable run forward under the driver seat up to the firewall and then make a 120-degree bend to point back towards the brake pedal. Similar to the system employed on most bicycles, the cable runs inside a sheath, which protects it and prevents it from deforming. Both the sheath and the 120-degree turn presented sizeable hurdles in making the braking system smooth and effective. The sharp turn is a substantial source of friction in the line, especially when subjected to the 50-100 lbs of braking force applied on the pedal through the cable. There is also internal friction between the cable and the sheath. This resistance increases the load requirement of the linear actuator, and makes it harder for the brake to return to its normal, unapplied position. To aid in the brake pedal's return to off, an extension spring was attached which provided a restoring force against depressing the pedal. This spring is attached at all times, and does make it slightly harder for a human to depress the pedal. In order to reduce the friction at the tight bend, a set of pulley blocks were custom manufactured using ball bearings to support the cable through the turn. The friction inside the sheath was hard to overcome. It requires that a layer of liquid Teflon lubricant be applied liberally throughout the sheath. It was also important to mount the pulley block appropriately. Much care was taken to ensure that the majority of the cable's pull was perpendicular to the pedal's direction of motion. This maximizes the force component from the cable which is ultimately applied to the pedal.

2.3 Emergency Brake Application. Although the main brake module was powerful and agile, competition rules as well as safety guidelines dictated the installation of an Emergency braking system. Several options were explored for the implementation of the emergency brake on the Canyon. One possibility included mounting a separate actuator onto the vehicle's parking brake. Since the emergency braking system should have a minimal stopping time, it was decided to have the emergency brake actuate the main brake pedal, and thus take advantage of a braking force on all wheels. It was also critical that the main brake module be kept independent and fail-safe module, so that in the event of a complete computer crash or electrical failure, the truck will positively stop.

With these considerations in mind, the design of the emergency brake evolved as follows. A pneumatically activated shuttle exerted tension on a cable that depressed the main brake pedal. This cable was completely independent from the main cable, running in parallel through the pulley block, and connecting to the brake pedal. The Teflon-lubricated sheath design was also used. The pneumatic system was custom designed to exclusive meet the needs of the emergency brake. A pneumatic cylinder with a 4-inch stroke was selected to match the maximum depth of the brake pedal. The cylinder has a 2.5-inch bore. To achieve the desired 150 lbs of force on the brake pedal, a system pressure of 30 psi is required. This pressure is supplied by a Campbell-Hausfeld oil-less compressor. This compressor stores pressurized air in its 2-gallon tank. The pump cycles the tank pressure between 70 and 100 psi. The output of the tank is regulated so that 30 psi is constantly supplied to the system, regardless of tank pressure.

To achieve fail-safe application, it is necessary that the emergency brake be active, even under its own failure. A series of one-way valves were installed to preserve positive pressure downstream. The compressor can literally be disconnected, and enough pressurized air remains trapped in the rest of the system to pressurize the cylinder and stop the vehicle. To minimize leaks after the one-

way valves, all tubing is surrounded by a plastic sheathing for protection. Also, failsafe operation requires that a positive signal to maintain a non-application. This is accomplished using a single-solenoid control valve for the cylinder. This type of valve has a default position, and an override position. Normally, it ports the pressurized air to one of its outputs. It takes a positive electrical signal to the solenoid to flip the valve and port the high-pressure air to the other output. The air lines were hooked up such that the default position of the valve causes the cylinder to fire, and depress the brake pedal. The valve is equipped with a manual switch to force it to the override position; this is useful during normal operation of the vehicle, in “manual mode.” In “autonomous mode” however, it is necessary that the computers be on to supply the electrical signal to keep the e-brake off.

In addition to the tension-sensor on the main brake line, the other important sensor needed for intelligent operation of the braking system is a brake pedal switch. This is a switch that comes standard with any vehicle, and is what dictates whether or not the brake lights are on. Our braking used this switch as an indication of whether the brakes were depressed or not. This is necessary because the tension sensor is often too noisy to determine a precise tension at which the brakes first came on. Additionally, if the main brake is off, and the car’s computer detects that the brakes are depressed. This is an indication that the emergency brakes has been applied, or that some other malfunction has occurred.

2.4 Steering wheel. The first mechanical system to come online during Prospect Eleven’s development was steering. This proved to be the most challenging of the systems to develop. The speed decision is relatively simple by comparison. Often during testing of the collision avoidance capabilities, it is more efficient to have a human modulate the speed of the vehicle while observing its steering behavior. This provides another reason that each system maintain its human-machine functionality as well as adopt a digital-mechanical capability.

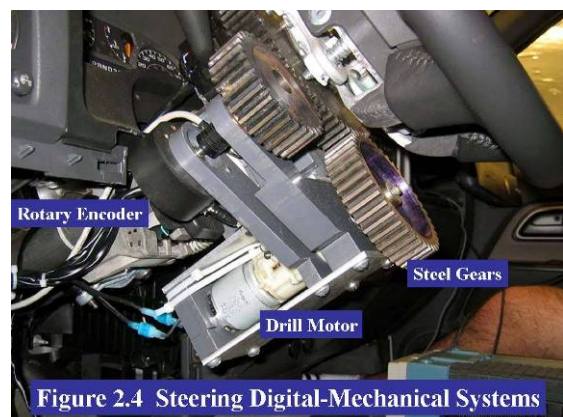


Figure 2.4 Steering Digital-Mechanical Systems

The steering actuator must be both fast and powerful, two qualities that are usually inversely dependent. Since the Canyon came with power-assist steering, less power would be required if the digital-mechanical system simply interfaced with the steering wheel. Several locations for the interface were considered before deciding to mount the steering actuator on the steering column just behind the steering wheel.

To mount the steering actuator, the steering column was disassembled. The airbag was removed as well as the feed through assembly for the electrical connections to the wheel. This created sufficient space behind the steering wheel and the turn-signal unit to mount 6” diameter steel gear on the steering shaft. Additionally, several mounting holes were discovered on the underside of the steering column, which provided a suitable attachment point for a motor housing.

The primary actuator of the steering system is a motor and gearbox extracted from a Ryobi 18V cordless drill. The gearbox is locked in the lower of its two gears, producing roughly 150in-lbs of torque at 200 rpm. This motor/gearbox assembly was selected because of its industry-proven

durability and efficient gear reduction. To maintain efficiency, the only external gear reduction is a 2:3 reduction between the motor output and the steering wheel. This reduction enables the steering wheel to spin at a maximum of 130 rpm, or about 2 turns per second, under a maximum torque of 225 in-lbs. This setup proved to be sufficiently fast, powerful and reliable.

The drill motor and gearbox are both mounted in a custom-fabricated housing which connects to the mounting holes on the bottom of the steering column. Extreme care was taken in aligning the large gear mounted on the steering wheel with the smaller gear mounted on the output of the motor. Extensive calibration and tuning was necessary to ensure that these gears turn smoothly and do not bind up through the entire rotation of the wheel. Human drivability is maintained due to the ability for a person to easily back-drive the drill motor. During human steering, the gears and motor are spinning, however they provided relatively little resistance in comparison to the normal torque needed to steer the vehicle.

The precise angular position of the steering wheel is obtained through a high-resolution rotary encoder connected to the large gear, see Figure 2.4.

2.5 Transmission shifter. Though the Canyon came with an automatic transmission, eliminating the need to individually shift gears, there still exists the desired to shift directions; from forward to reverse. To do this, it is necessary to actuate the transmission and shift it from Drive, into Reverse. Drive and Reverse are not adjacent positions on the Canyon's transmission, and shifting between then requires going through Neutral. Since Neutral and Park are also desirable positions to obtain, it a linear actuator was designed and built to manipulate the transmission.

On the Canyon, the transmission shifting lever is normally mounted on the right hand side of the steering column. This lever connects to a stiff cable which runs under the floorboards to the transmission. One possibility for actuating the transmission was by attaching an actuator to the physical transmission, on the underbody. This was disregarded because of its perilous location. The mounting of a linear actuator on the steering column proved to be a much simpler approach. The normal shifting lever is held in by three mounting screws which provide a solid base on which anchor a linear actuator once the shift lever is removed. Since mounting the actuator on the column necessitated the removal of the manual control, it became essential that the actuator have human-controllable capabilities. Thus an actuator was designed to include a mechanical knob to enable manual-mechanical operation, a push-button configuration to enable manual execution of the electro-mechanical assembly and an electrical connection that enabled digital control of the gear shifter.

The linear actuator which shifts the transmission was completely computer designed using Autodesk Inventor. The actuator required the precise interfacing of a motor, a lead screw, several bushings, to mounting holes on the steering column. Having everything laid out in virtual form made the fabrication and installation process much easier. The actuator was fabricated out of stock materials purchased from McMaster-Carr Supply Company, and required roughly 20 hours of machining and assembly.

Ultimately, the transmission shifter was not used during the Grand Challenge. Its slow actuation presented difficulty for normal operation of the vehicle. Additionally, it was decided that allowing Prospect Eleven to go in reverse could introduce more problems than it solved. The possibility of

false-positives initiating an unnecessary reverse command was considered more risky than the probability of encountering a dead-end and needing to go in reverse; however, successful implementation of a digital-mechanical shifter is one of the most important elements of future work on Prospect Eleven.

2.6 Other digital-mechanical tasks. In addition to designing, building and mounting the digital-mechanical systems, the student team was charged with mounting and protecting the major sensors for Prospect Eleven. Prospect Eleven navigated through the use of only two primary sensors. One was a stereo camera, and the other was the multitude of GPS antennae for position information.

The stereo camera, a Bumblebee made by Point Grey Research Inc. (see section 3) is nominally rated for “indoor office use only.” However, to maximize its useful range, it was positioned at the very front of the hood. This created the need for a water- and air- tight enclosure which could protect the camera from the harsh desert environment. This enclosure was fabricated out of PVC plastic. PVC was chosen for its ease of machining, its slight deformation properties, and its attractive cost. After being computer designed in Autodesk Inventor, all of the sides for the enclosure were then hand-milled or CNC-machined to very tight tolerances. The sides were then assembled together around the camera unit. The camera unit itself is mounted on a small block of aluminum for heat absorption.

The stereo camera contained black & white CCDs. To enhance the image quality, red photographic filters are needed to increase contrast and darken much of the sky. Additionally, UV filters are used as a front layer of protection for the red filters. Standard step-ring adapters for the filters are securely mounted to the front-plate of the camera enclosure. This allows any combination of red, UV, and even orange filters to be mounted in front of the camera. As long as at least one filter is tightly fitted, the housing maintains its isolation properties.

Prospect Eleven calculates its position based on the information received from four GPS antennae. Two are Trimble GPS units, which feed their information into an Applanix POS-LV system, which uses an additional two antennae. All of these antennae are mounted on the roof of the vehicle so as to have the most unobstructed view of the sky. The POS-LV system’s two antennae are the primary ones used to calculate uncorrected GPS position; the system receives differential corrections from the two auxiliary GPS units. The POS-LV also uses its two antennae to calculate heading. Using a procedure called GAMS (gps azimuth measurement system), the system uses carrier-phase differencing between the two antennae to calculate the heading of the vehicle. It is necessary that the two main antennae be separated by a known distance of at least two meters. To maintain the specific positioning of the GAMS antennae, as well as the other GPS receivers, they are all mounted on a roof-located boom. This boom was constructed from an eight-foot length of 2x4 lumber. Each GPS antenna was given a specific post on the boom, with each post secured through the boom at known intervals. The boom also served as a suitable mounting surface for the rotating beacon light and the antennae from the DARPA-supplied E-Stop unit

The final task for the Vehicle Mechanics team was the hardening and protection of the remainder of the physical systems, especially the computers. Everything on the vehicle has to be able to

withstand the bumps and jolts associated with traversing unimproved desert roads at moderate speeds. This creates a harsh environment for the computer systems and associated hard drives.

Prospect Eleven relies on two computers, one for vision processing and one for navigation decision. Both computers are mounted in a standard rack that is secured to the vehicle on shock-absorbing feet donated by Shock-Tech. These feet consists of a neoprene rubber shells filled with silicone gel. Each of the four feet is rated for 30 lbs shock absorption. The weight of the steel rack plus the two computers was just over 100 lbs.

3. Obstacle Detection Using Stereo Vision.

Among Prospect Eleven's most distinctive features is its reliance on stereo vision. Indeed, it was the only vehicle in the Grand Challenge event which used stereo vision alone. This section discusses the challenges of using stereo vision for an Autonomous Ground Vehicles, describes the hardware and algorithms Prospect Eleven uses to detect and track obstacles, and considers future work in the field.

Stereo vision, the process of converting two simultaneous images from synchronized spatially-separated cameras into a depth image, is a well-studied problem. The key challenge is to reliably determine the correspondence between of the features of an object as they are captured on each of the two images. The separation distance between the corresponding features on each of the images is proportional to the distance. A depth image, or disparity map, is the ensemble of the distances of all corresponding features in a scene. Several vendors sell systems which include synchronized cameras and SDK that can produces a disparity map tuned to several parameters. Purchased was one such system, known as a Bumblebee™, from Point Grey Research (PGR). It contains two black and white CCD cameras at a 12 cm baseline separation. The PGR Software Development Kit (SDK) produces a disparity map at a rate of approximately 16Hz from synchronized images. Given the disparity maps, we focused on the problem of obstacle detection and tracking given the capability to generate a range of disparity maps as a function of several parameters.

Stereo's reputation for producing noisy results is well-deserved. Data is heavily quantized due to pixilation of the image— that is, a point can take on only a small number of possible depth values. Moreover, in areas of low texture, correspondence of features can become very unreliable. Though the PGR SDK had fairly robust validation routines and would generally not report false matches, many environments generated sparse disparity maps. Lighting conditions also present a problem. Images with shadows, for instance, are frequently either excessively dark in the shadowed region or are washed out in areas the lighted region. These are issues that can not be ignored. Our efforts to deal with them fall into three main categories:

1. Ensuring that images of the scene have sufficient contrast to generate dense disparity maps,
2. Using obstacle detection algorithms which are robust to noise and can take advantage of quantization, and
3. Tracking known obstacles in the time domain.

3.1 Generating disparity maps. Several strategies proved effective for improving the quality of scene images. Red and UV photographic filters, mounted in front of each lens, help increase contrast and remove specular features. In particular, red filters helped reduce the intensity of the

sky and sun, preventing issues such as CCD “bleeding.” Though the CCDs have their own autogain control, it seems to be tuned to generate a contrast level more appropriate for human viewing than for disparity processing. Fortunately, the PGR SDK allows the camera’s gain to be controlled in code. By experimentation, it was found that the best depth maps were generated by relatively dark images. Implemented was the following simple control law to govern the camera’s gain:

$$G' = G + k(C - T),$$

where G' is the new gain at a given iteration, G is the current gain, k is a gain term, C is the current average intensity value sampled over some region of interest, and T is the target intensity value.

This simple control law is far from the state of the art in control theory. The camera was sometimes slow to adjust to sudden changes in brightness such as a transition in and out of a dark tunnel. However, it was adequate for nearly all situations Prospect Eleven encountered. The combination of photographic filters and improved control of the camera’s gain dramatically improves the range of situations in which the PGR SDK can generate full disparity maps.

Despite these improvements in image quality pre-disparity matching, problems with the depth images remain. Ideal lighting conditions do not guarantee the accuracy of the correspondence matching throughout the image plane. Fortunately, the PGR SDK’s validation routines are quite robust, and tend to reported only reliable matches. Thus, the disparity maps can at times be quite sparse; however, one can reliably assume that data reported is accurate, at least to the extent possible within the constraint of heavy quantization.

3.2 Obstacle detection Given the disparity map, the problem at hand was the detection of obstacles in the field of view. Implicitly, it was assumed the viewed landscape consisted of a plane surface with substantial “obstacles” seated on that surface extending above (and possibly below) that ground plane. The implication of such a simplified view of the world is that an obstacle free surface would exhibit a map whose disparity would monotonically increase as one moved up the map and be constant across the map. Obstacles perpendicular to that surface would exhibit constant disparity throughout. With this in mind, Prospect Eleven used the following algorithm to isolate obstacle:

- 1) FOR each column in the disparity map
 - a) Consider a DFA with states { IN OBSTACLE, NOT IN OBSTACLE }
 - b) Begin in state NOT IN OBSTACLE
 - c) FOR each pixel in the column, starting at the top of the disparity map
 - i) IF state is NOT IN OBSTACLE
 - (1) Consider the next m pixels. If they are all the same, transition to state IN OBSTACLE
 - (2) Consider the difference, disparity for the next pixel minus disparity for this pixel. If this is greater than k_D , transition to state IN OBSTACLE
 - ii) IF state is IN OBSTACLE
 - (1) Calculate the variance over the next m pixels.
 - (2) IF this quantity is greater than k_V , transition to state NOT IN OBSTACLE.

- (a) Calculate the variance over the last span of pixels for which the DFA was in state IN OBSTACLE.
- (b) Add this span, and its variance, to a list of obstacles for this column

(This algorithm is a bit simplified, as the actual implementation includes logic to deal with invalidated pixels. However, this logic greatly complicates the algorithm without adding to the discussion.)

It is worth discussing the decision to process on disparity maps directly. As [1] notes, processing a disparity map is substantially faster than working on an elevation map (as does [2]), and maintains the highest possible resolution of data.

Several properties of the simple column-detection algorithm are advantageous. First, it is robust to sparse disparity maps and high quantization. Because it does not rely on any global characteristics of the image, such as the computation of a global ground-plane, sparse disparity maps do not greatly inhibit performance. Instead, two conditions are sufficient: an abrupt change from background to a foreground object, or an object which is (approximately) parallel to the image plane. The algorithm takes advantage of heavy quantization, as objects which are approximately parallel to the image plane will have the same disparity value throughout.

There are trade-offs to the simplicity of this algorithm, however. Because distance is inversely related to disparity, we should expect that vertical bands of the same disparity value will grow smaller as disparity increases. The constant m assumes that this value remains constant. Approaches like [1] do not suffer from this problem, but are slower as a result. Figure 3.1 below provides an example of a scene, the corresponding disparity map, the generated columns and the resulting bounding rectangles representing the obstacles.

This simple algorithm runs in time linear to the number of pixels. At the termination of the algorithm, a list of row-spans in each column is classified as obstacles. The variance serves as a measure of the confidence in an obstacle's existence. A local search algorithm bounds these spans with rectangles, ensuring that there exists a degree of uniformity across columns. This step also helps with much of the noise in the disparity map, as a great deal of noise does not exist in several contiguous columns. Once a bounding rectangle is determined, points in the rectangle are ranged, and averaged. Assigned is an (x,y) location for each obstacle relative to the nose of the vehicle.

3.1 Filtering in the time domain. Unfortunately, these range measurements are not very accurate. [3], for instance, refines depth estimates using LIDAR. To make them useful for an autonomous ground vehicle, filtering in the time domain is crucial. In particular, a list of all obstacles of concern to the collision avoidance routine is maintained. Obstacles isolated by a new frame are compared to the list of existing obstacles. Each new obstacle is matched to an existing obstacle, or declared to be a new obstacle. If it is matched with an old obstacle, the position is updated as a linear combination of the old and new position. In addition to providing dramatically increased accuracy in positioning, this allows the removal of many false positives, as their randomness tends to not have them appear in the same location in multiple frames.

By realizing that the scene ahead is rather simple, enhancing image quality, utilizing tracking in the time domain to build confidence and structuring algorithms which are robust to the errors

stereo frequently encounters we have demonstrated that stereo vision is a feasible platform for mid-speed autonomous ground vehicle navigation. Its primary limitation is range. This substantially limits prospect Eleven's maximum speed in the presence of centerline obstacles. In its current state, speeds, in the presence of unexpected obstacles was limited to approximately 25 mph. Ongoing research by the authors aims to extend stereo's range, such that it will be a suitable sensing platform for high-speed navigation.

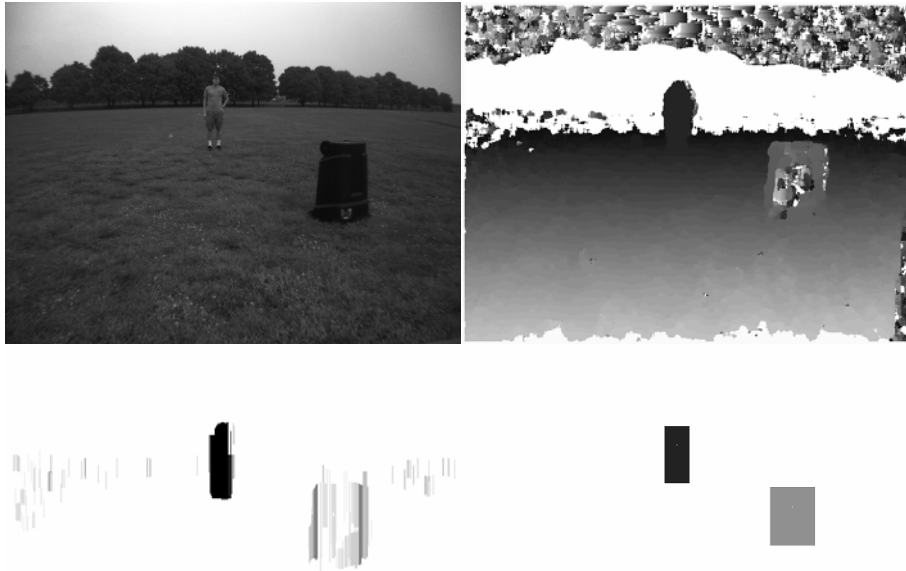


Figure 3.1 Example of stereo vision algorithm. Upper left: Original image; Upper right: Disparity map; Lower left: Processing in columns; Lower right: Bounding with rectangles

4. Software framework and computing systems

All computing is performed by two standard desktop computers. Each computer is rack-mounted in a 4U case and installed in a shock-isolated rack. One computer, named Santiago, is based on an Intel Pentium 4 processor and the other, called Prospero, is based on an AMD Athlon 64 processor. Vision processing and obstacle detection algorithms are written in C++ and run on Prospero, due to its strong ability to handle such computations. The drive-by-wire, data acquisition and decision making control systems run on Santiago, whose processor was selected due to its "HyperThreading" capabilities.

The software framework running on these computers is responsible for dealing with sensor inputs, obtaining the output of the navigation algorithm, and finally commanding the vehicle to drive. It processes the information coming from the vehicle sensors, directs it to the appropriate decision-making subroutine, assembles the result of these subroutines and engages the vehicle's control systems to enact the desired result. This code is written in C# ("C-sharp"), a programming language recently developed Microsoft. C# was chosen over other potential languages (such as C++) because it is extremely flexible, reasonably fast (relative to similar languages such as Java), and through its integration with the .NET framework allows native access to a wide variety of

Windows API functionality. Most importantly, however, it is very easy to use and comes with a highly-refined integrated development environment (IDE) allowing us to focus primarily on code operation and implementation rather than language syntax or low-level computer control. In addition to these benefits, C# compiles to “managed code.” This means that in effect the routines run in a virtual machine, protecting the system at large from being crashed by a programming error. This flexible error-handling scheme is a significant advantage of C#. The control routines take advantage of its ability to prevent buffer-underrun errors, file and network I/O errors, and other such issues from being fatal to the entire program or the operating system itself.

The basic approach in the design of the software framework was to make the code multi-threaded, so that each component of the code is able to run independently of the other components. Communication between these different components is achieved primarily using the C# support for events. An event in C# consists of a list of functions requesting to be notified when the event in question is triggered. This occurs when relevant information is obtained about the state of the vehicle, such as new position information from the GPS system. The functions that are to be notified when an event is triggered are structured to receive a class containing the relevant data about that event (the “EventArgs”). Such a function “hook” can be added or removed from the event at any point during the program.

The event-based software framework is implemented as follows: Upon initialization of the program and the relevant vehicle devices, each class of the control code inserts a hook into the events it cares to be notified about. In the case of the position determination algorithm, for example, a hook is inserted into the “GpsRmcSentenceReceived” event, which then causes the control code to be notified each time a NMEA 0183 sentence containing position information is received from the GPS system.

The event-based structure for the vehicle framework has a number of advantages. First, it is compatible with the asynchronous nature of the sensors, since there is no correlation between the timing of GPS position, for example, and the timing heading information from a digital compass. Thus the latest available sensor information is always available without polling. This of course means that the asynchronous nature of the sensor data must be handled in the classes that use the information from the sensors rather than in the sensor communication code.

The second advantage of the event-based approach is that it simplifies monitoring of the vehicle’s systems. It is very easy to create another component that simply inserts hooks into all the devices and simply monitors the data received to ensure the proper working of the various devices. If this monitoring code determines that the compass has begun to malfunction, for example, it is able to trigger another event which notifies downstream code to use an alternative source of information. Thus, event-based coding aids in self-diagnosis of a problem as it occur which substantially improve the code’s reliability.

The third advantage is simply both cosmetic and fundamentally valuable. Event-based coding greatly simplifies the overlay of graphical user interfaces (GUI). This facilitated us to much more readily calibrate and debug the various components. More importantly, since the GUI can be readily switched on and off in the event-based framework, it consumes essentially no resources when turned off. As a result, the GUI causes no performance loss whatsoever when it is not running.

Another fundamental aspect of the code is its modularity. From the ground up, the framework is designed so that each class is as independent as possible, referencing only those components that it needs to operate. Under this structure, each component of the system, whether it is a class to communicate with the motor controller, a thread to control the steering wheel position, or the genetic navigation algorithm, compiles to its own library. These libraries are then referenced by other classes as necessary.

Probably the greatest advantage of this structure is the ease with which programs can be made to develop and test the individual components of the system. When writing the speed control algorithm, for example, a new project was simply created to contain the new algorithm and referenced the existing brake and throttle control libraries. These libraries in turn reference the motor control and data acquisition libraries, respectively. Thus, to control the throttle and brakes required only: “Throttle.Position = x” or “Brakes.DesiredTension = y.” In addition, once the speed control code was completed, the direct control the speed was accomplished by writing “Speed.DesiredSpeed = z.” This modular approach to system design has been one of the hallmarks of our vehicle framework.

Figure 4.1 below depicts a block diagram of our computing systems:

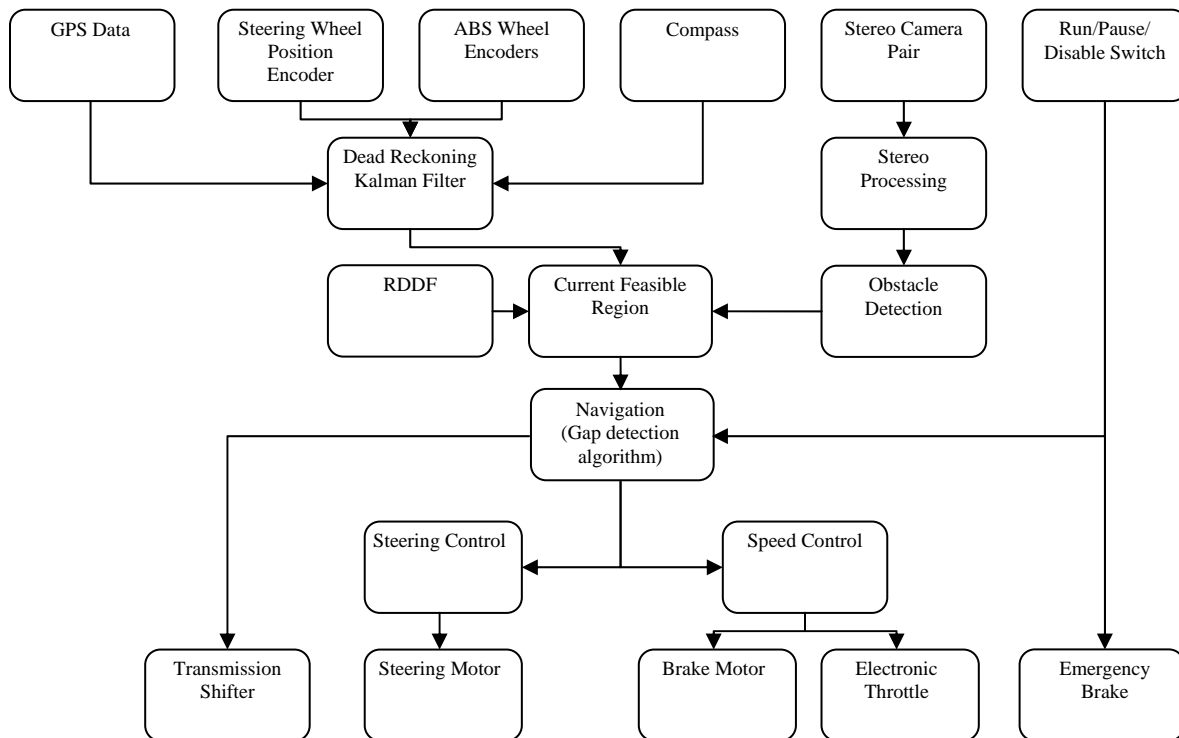


Figure 4.1: System block diagram

4.2 Next Steps The computing system was designed and implemented based on the system with stated goal of producing the most elegant system possible by trying to maximize flexibility, productivity, and reliability. Nonetheless, the system is not yet complete, and it is possible to improve it in a number of ways. Yet to be implemented is an overarching system monitor module, which would ensure that all system components are functioning as expected, provide alternatives

if necessary, and continuously log all data for analysis. Under investigation are a number of ways to improve sensor fusion, including the use of an Extended Kalman Filter, which should help to better arbitrate when various sensors provide conflicting information. Despite the work left to be done, a simple yet robust autonomous vehicle computing system has been developed that is a platform on which one can build.

5. Vehicle Guidance and Control

In many ways, Prospect Eleven's guidance system mimics the actions and decisions of a blind person. At each instant of time, it probes only a limited region ahead to find, within this limited region, the best collision-free path ahead. Once found, it directs its control systems to seek the path, collects updated information about the new current limited region ahead and returns to find a new current-best collision-free path. Each loop takes roughly one-tenth of a second to complete and returns only two scalars, desired heading angle and desired speed. As such, Prospect Eleven evolves autonomously through the course making the best decisions it can with the latest available information about the evolving limited region ahead. While certainly not globally optimal, the simplicity of this fundamental approach contributed substantially to its successful implementation.

5.1 The limited region. The best collision-free path is found within a standard yaw-pitch-roll moving coordinate system defined by the instantaneous ground-plane, steering axis and center of the stereo vision system. Lane boundaries and other GPS based information is mapped to the moving coordinate system based on the best GPS/INU global state estimates at the time of the latest image analysis of the stereo camera system. Image analysis from the stereo cameras directly locates identified obstacles in this coordinate system. Polygons that specify the obstacles and lane boundaries identify the infeasible areas within the limited region. Furthermore, since the stereo system has a limited range, the steering system has a limited turning radius and rollover dynamics limit the steering response at "higher" speeds, the limited region is further bounded by a forward cone. It is within this instantaneous feasible region that the guidance system searches to find gaps in the feasible region and choose a value for steering angle and speed.

5.2 Finding gaps in the feasible region. A family of simple feasible paths ahead can be characterized by virtual linear "tubes" whose individual width is slightly greater than the vehicle and cast from the origin of the moving coordinate system through feasible values of yaw. The extent of the feasibility of each path is the nearest intersection with an infeasible boundary. Figure 5.1 depicts such a family of paths as well as a plot of their path length as a function of yaw angle.

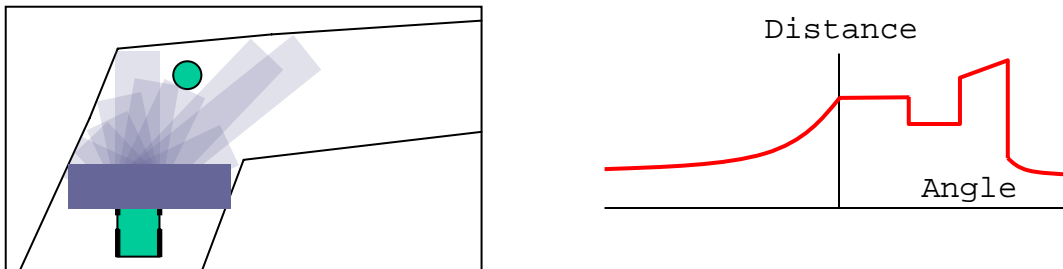


Figure 5.1 -- Tube Projection and Polar Plot

By tagging each tube with the boundary that forced truncation, one can see that changes in boundaries tend to introduce second-derivative-type jumps in the tube length plot. Pairing of these jumps tends to identify gaps between obstacles and boundaries in the feasible region ahead. Figure 5.2 shows the three discontinuities identified in a discrete version of the plot in Figure 5.1. Colors represent tubes terminating in different sorts of obstructions.

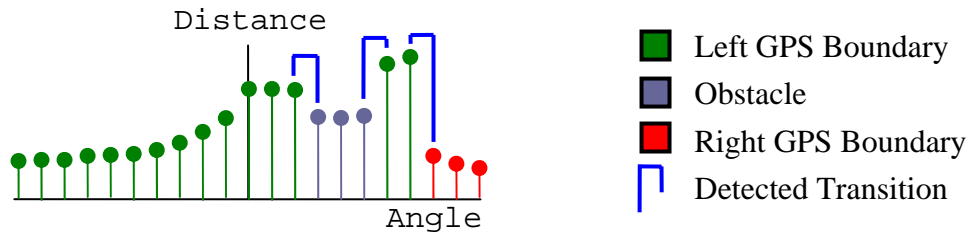


Figure 5.2 -- Discontinuity Detection in Discrete Plot

Extraction of gaps is achieved by scanning from the higher side of each transition with a jump greater than the car's width until reaching either a distance less than the lower side of the transition or the higher side of another transition. The two cases are demonstrated in figure 5.3. When comparing distances, each distance is weighted by the cosine of the angle it makes relative to the car's heading. This weighting forms gaps perpendicular to the car's direction of travel.

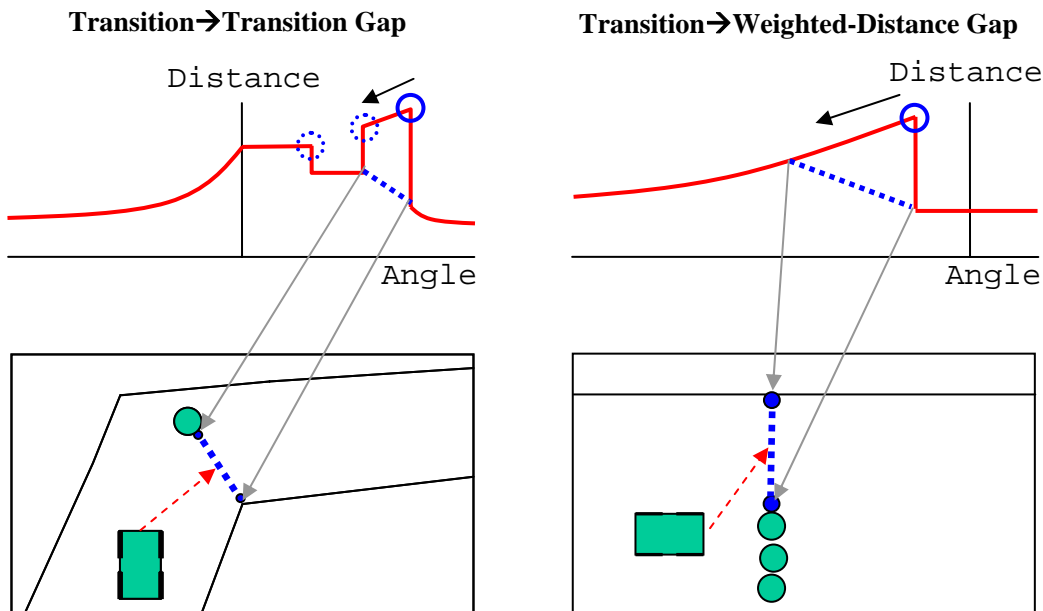


Figure 5.3 -- Gap Computation

5.3 Choosing a desired steering angle. For each gap ahead a center point is identified to provide a feasible heading vector defined by a yaw angle and length. Furthermore, a tube is cast to an ideal pursuit point located along the GPS course centerline at a fixed distance ahead proportional to the desired speed of the vehicle. If this tube does not interact with any obstacle or boundary, then its corresponding heading vector is added to the set of feasible heading vectors. Choice among the feasible heading vectors is based on length and the change in heading angle from the heading angle chosen in the previous iteration. If the ideal pursuit point is feasible the angle

change is small, then it is chosen; otherwise the longest tube with the smallest angle change is chosen. In some cases this becomes an extremely delicate balance between these two elements. The desire is to avoid oscillations of the heading angle while continuing to seek the longest unobstructed path. Much of the “art” of the navigation system is contained within this tradeoff. The yaw angle of the heading vector is fed to the steering control system for implementation.

5.4 Setting desired speed. The desired speed fed to the control system is the minimum of a set of values that includes a global maximum vehicle speed limit, a local segment speed limit as specified by the in the RDDF, and a speed limit proportional to the curvature of the GPS centerline ahead and the length of the instantaneous best heading vector. The minimum of the global speed limit, segment speed limit and curvature speed limit defines a desired speed limit which is converted into a pursuit length. Current instantaneous speed defines a minimum stopping distance. The ratio

$$\frac{(\text{heading vector length} - \text{min stopping distance})}{\text{pursuit length}}$$
 modulates the desired speed between zero (stop command) and desired speed limit. This desired speed is fed into the speed control system for implementation.

5.5 Limitations of guidance algorithm. The output of the algorithm, shown in figure 5.4, handles many cases well. Its commands can be sophisticated, as when it commands the vehicle to drive at a row of obstacles because it knows a gap is available.

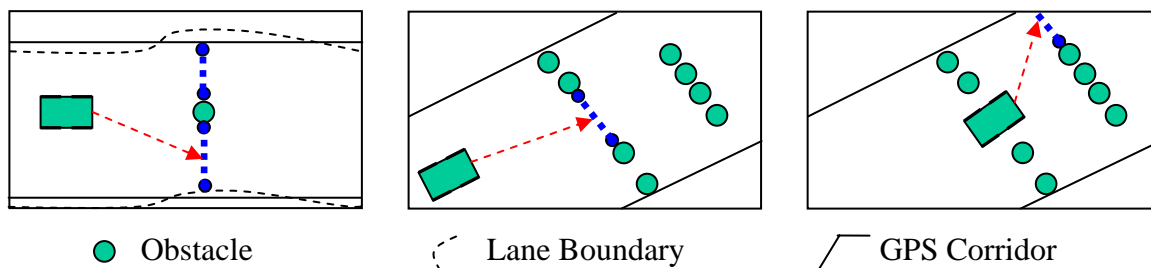


Figure 5.4 -- Gap Identification Algorithm Output

But, the algorithm is local and thus can become stuck in certain configurations. As shown in figure 5.5, if tubes can project into a dead-end it will not appear as a viable gap. If tubes cannot project into the dead-end, then it is possible that the vehicle will drive towards it. A critical feature of this algorithm, however, is that upon advancing to the dead-end, the algorithm will unambiguously report no available gaps. The vehicle can thus, at the very least, stop safely.

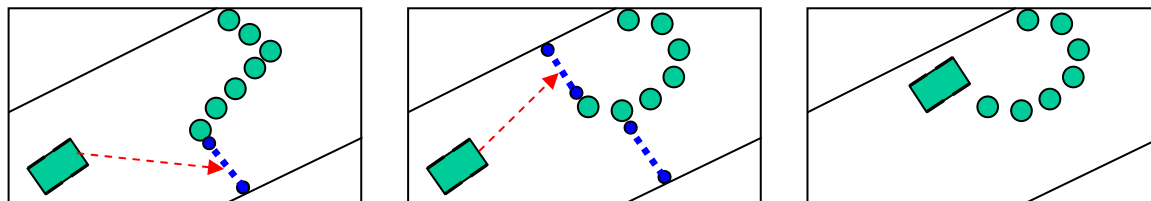


Figure 5.5 -- A Problematic Case

5.6 Vehicle Control. Implementation of the desired speed and heading angle are accomplished in the vehicle control loop. Output are control commands to the vehicle's brake, throttle and steering mechanisms. The objective is to gracefully achieve desired speed and heading angle; that is to do it smoothly, while responsively without a great deal of overshoot and to quickly implement any emergency braking command. A standard proportional-integral (P-I-D) feedback control loop was implemented for both the speed and steering controllers, see Figure 5.6. For the speed control, vehicle guidance provides the desired speed and the feedback loop provides current speed. Output are percent throttle or percent brake which are fed directly into the brake and throttle digital-mechanical systems. For steering control, desired heading angle is converted into a steering angle using Nichols-Ziegler closed loop turning equations. Current steering angle is provided by the feedback loop. The P-I-D control loop provides the desired the steering angle that is fed to the steering digital-mechanical system.

Speed Control -- Implementation

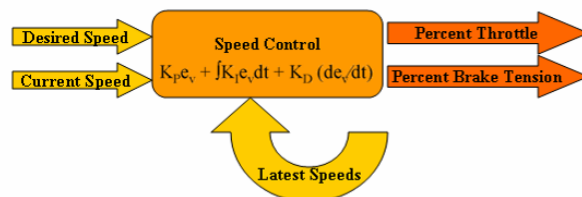


Figure 5.6 – PID Speed Control Loop

6. Accomplishments

6.1 Pre-National Qualifying Event (NQE) Prospect Eleven's road to the 2005 Challenge was fraught with "do-overs" that cycled between failure and success. The vehicle failed the first site visit during May 2005, only to be given a second chance as an "Alternate". An essentially flawless performance on the three required runs and a fourth optional run earned Prospect Eleven an invitation to the National Qualifying Event (NQE) during its second site visit in August. Linked are depictions of the GPS [data](#) of the [1st run](#), [2nd run](#) and [3rd run](#). Layout of the [optional fourth run](#) and a [video](#) of that run are provided as links.

6.2 Pre-National Qualifying Event (NQE) The NQE involved the running of a [2.2 mile closed serpentine course](#) defined by GPS waypoints with associated lateral boundary off-sets and speed limits that were to constrain the "bots". Fifty (50) "gates" marked some of the boundaries and five (5) obstacles (four (4) vehicles and a Normandy-style tank trap) were placed in the "center" of the desired travel lanes. The course also included a 100 foot-long tunnel (no GPS availability), rumble strips, hay-bale-lined boundaries and a concrete barriers simulated some of the narrow and rough conditions that were to be expected during the Grand Challenge. At the NQE Prospect Eleven made five runs of the 2.2 mile qualifying course. It performed spectacularly in each odd-numbered run, and disastrously in runs 2 and 4. In fact, its first run may have been the best run of any vehicle at the NQE. Not until after the 3rd run was it realized that Prospect Eleven's GPS array was misaligned. This caused a biased shift in its perception of the GPS course boundaries. This bias, when combined with the placement of the physical gates meant that many of the gates were perceived to be obstacles within the course boundaries rather than physical markers of the

course boundary. Thus, the autonomous path perceived to be available was substantially narrower and had more obstacles than the actual course. The fact that it successfully completed that run, while only nipping two gates and avoiding all other obstacles, was truly remarkable.

The GPS alignment problem and a sluggish response of the braking system caused crashes with the first gate and a parked vehicle and an early termination of the second run. Prior to the third run improvements were made to braking system's dynamic response and, suspecting bias in GPS, an offset translation was made in the GPS code. This kluge allowed Prospect Eleven to successfully traverse all 50 gates and avoid all 5 obstacles; however, it did nick a couple of the tire stacks.

The 4th run ended in failure due to instability in the steering controller. It turned out that two inappropriate remote processes had been left running on Prospect Eleven's vision computer. Not long into the run, these processes consumed essentially all of the computing resources, such that the vision computer was processing less than one frame per second, much too sluggish to remain stable.

For the 5th run, "everything" was fixed and Prospect Eleven had an essentially perfect run. It earned a 10th seed in the Grand Challenge Event.

6.3 Grand Challenge Event (GCE) Twenty-three (23) bots were invited to participate in the Grand Challenge Event on October 8, 2005. Prospect Eleven was seeded 10th. The event involved the traversal of 132 mile course in the desert outside Primm, Nevada. The fastest bot completing the course in less than 10 hours would claim the \$2 million prize. The actual coordinates of the course were not divulged to the teams until 4:00 am on October, 8; 2 hours before the start of the event. Provided was a standard file containing waypoints, lane widths and speed limits. No information was provided on lane roughness nor the location of obstacles, if any. In preparation for the receipt of this file, Josh Herbach'08 had written an analysis program that allowed us to view, evaluate and modify the course. This allowed Josh, Andrew Saxe'08 and I to modify the constraints and to estimate some of the impacts of the modification. While we had conducted one high speed test of Prospect Eleven, we were not confident about its performance at high speed. Since a 35 mph global speed limit allowed for completion in just under 9 hours, we imposed this conservative constraint. Also, we were more confident in Prospect Eleven's stereo vision system than we were in the unbiased accuracy of GPS. Consequently we increased the narrowest lanes to be at least 9 feet wide, thus relying on vision to keep Prospect Eleven on a collision-free path ahead.

Prospect Eleven performed admirably in the Challenge. It launched without difficulty ([see video](#)) and reappeared on schedule at the 8 mile mark ([see video](#)) and passed over the Union Pacific mainline at the 9 mile mark. Unfortunately, the steering became unstable shortly thereafter at about 9.4 miles. A "Personal Best"! After recovery of the vehicle, it was apparent that there was a bug in the obstacle detection code that grew to consume essentially all of the computing resources. One segment of the code failed to dispose of obstacles that had been passed. By the 9th mile, the process was analyzing thousands of obstacles in its effort to determine a collision-free way ahead; thus, it could only update the steering system less than once a second.

In the end, Volkswagen (a.k.a. Stanford) won the Challenge by finishing in just under 7 hours.

Carnegie Mellon's two entries, Highlander and Sandstorm, finished 2nd and 3rd in just over 7 hours and two others, Grey Team and TerraMax completed the course in about 10 hours, but the students of Prospect Eleven were the real winners. The learning and experience were "Priceless". They knew in their hearts that they had created a phenomenal autonomous vehicle. The only disappointment was in not proving it.

6.4 Unfinished Business Shortly after the team's return to campus and the placement of Prospect Eleven on in its van "back to Nassau Hall" Bryan Cattle'08 called Wm. Culbreth, Dean of Engineering at UNLV inquiring about temporary storage for Prospect Eleven. The resulting enthusiastic offer of assistance made it clear that Prospect Eleven needed to be turned around in order to give this Prospect Eleven yet another "second chance" to prove its worth. There was unfinished business to take of and the upcoming Fall Break was the time to "Just Do It". So, on Saturday evening, October 30, 5 team members, Andrew Saxe'08, Gordon Franken'08, Bryan Cattle'07, Anand Atraye'07 and Scott Schiffres'06 and Prof. Kornhauser returned to Las Vegas. Joined by Ben Essenburg'05 on Sunday morning, they found Prospect Eleven at the [Center for Energy Research](#). Using the [shade cast](#) by the Amonix Integrated High Concentration Photovoltaic panel, Anand Atraye set out to find the bug in the code. In a couple of places in the code, he changed one line:

"Everywhere I saw something like:

```
currentObstacles.RemoveAt(i);
```

I preceded it with:

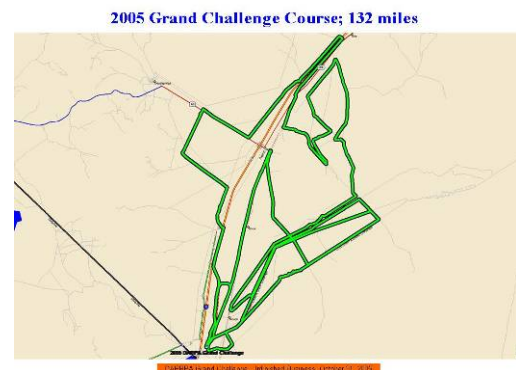
```
State.RelativeFrameUpdated -=  
currentObstacles[i].relativeFrameEventHandler;
```

The problem was that each time an obstacle was removed from the list of current obstacles, it was not being unhooked from the `RelativeFrameUpdated` event. Thus, the garbage collector never determined that the old obstacle was no longer needed, and as a result it was never cleared." In other words, Prospect Eleven never really "forgot" about an obstacle that it had seen and continued to determine if it needed to avoid it. Thus, after 9+ miles, it was still evaluating obstacles it had detected at the start. The list had grown to thousands of obstacles.

With the code change and a recalibration of the GPS/INU, Prospect Eleven was driven the 35 miles down I-15 to Primm in position for a second chance at the 2005 DARPA Grand Challenge on Monday morning.

6.4.1 Assault on the 2005 Grand Challenge Course.

Early Monday morning, October 31, 2005, ironically Halloween, we set out to run the 2005 Grand Challenge course exactly as we did during the actual Grand Challenge. Prospect Eleven was using the same RDDF (file of GPS waypoints that define the course) and the same global constraints and control coefficients. The only substantive difference was the change in the "one line of code". Since we had limited support personnel, comprising of two support vehicles, it was necessary to have someone ride inside Prospect Eleven in case an emergency stop condition was encountered.



It simply wasn't practical to monitor both the environment ahead and the stability of the vehicle entirely from the support vehicles. While possibly dangerous, Professor Kornhauser rode the entire distance behind the wheel of Prospect Eleven, prepared to instantly take over command of the steering and brakes, while in no other way interfering with its autonomous operation. While the plan was for Professor Kornhauser to be alone, Andrew Saxe was permitted to ride "shotgun" during the first stage which traversed a dry lake bed. Since Prospect Eleven had successfully driven this section during the actual Challenge, it was expected to pose few risks. Andrew's job was to toggle the "kill" switches that would enable Prof. Kornhauser to more easily take over control of the vehicle in case of emergency.

Launch came at 7:53:30PST and was uneventful. Everything was perfect until just a few miles into the course when a mirage seemed to appear in the distance. Not to worry, it's the desert; however, it quickly became apparent that the "dry" lake was not so dry. It had rained since the Grand Challenge and the course was not traversable in a non-amphibious vehicle. The decision was to cease autonomous operation in order to not lose the vehicle. A precise autonomous run of the 2005 GC course was infeasible because of the rain. With the current condition, no Grand Challenge vehicle could have made it beyond this point. In fact, if this condition would have existed during the Grand Challenge, DARPA would necessarily have had to alter the course. It now became evident why, during the Grand Challenge, the course was not divulged earlier than 2 hours before the race. It was to ensure that the course was a fair one and that some environmental condition had not made a part of the course impassable.

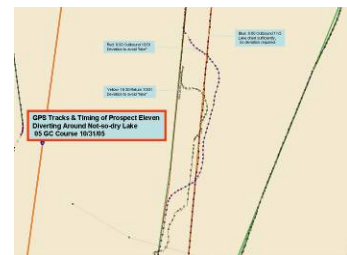
Rather than go home, the decision was to continue to uncover Prospect Eleven's autonomous operational limits by continuing on the traversable portions of the 2005 GC course. The first autonomous navigation limit had been established: it can't traverse lakes and isn't smart enough to figure out a way around them, if the "desired" course is through them. This has become one of the major goals of our future research.

After a brief diversion around the lake, autonomous operation was reinitiated at reemergence of the 2005 GC course. This incident made it apparent that two people were needed inside the vehicle to properly monitor the road ahead. Other than the lake situation (which occurred at 2 other points), the only non-autonomous diversions were due to

1. places where the "road" had been "bulldozed" probably to discourage exactly what we were trying to do. These places existed at 14:16 and 16:31 (See Figure 6.1), and
2. on NV 604, a public road, where we pulled over to let a cement truck pass us (if this situation would have occurred during the Challenge, DARPA would have paused the vehicle and instructed the cement truck to carefully pass the vehicle).

These two incidents define the operational limits of the current system. Specifically, Prospect Eleven needs the autonomous ability to violate non-critical route constraints and set out to find any feasible path ahead, and it needs to be able to deal with overtaking objects. At present, it does not have these capabilities.

Also, Prospect Eleven was paused several times, much the same way that DARPA may have legitimately paused the vehicle during the Grand Challenge. Pauses were instituted prior to crossing public roads, the Union Pacific at-grade crossing, upon encountering closed gates, that



once opened, were negotiated autonomously and for preparing the onboard camera to record the traverse of Beer Bottle Pass at night.

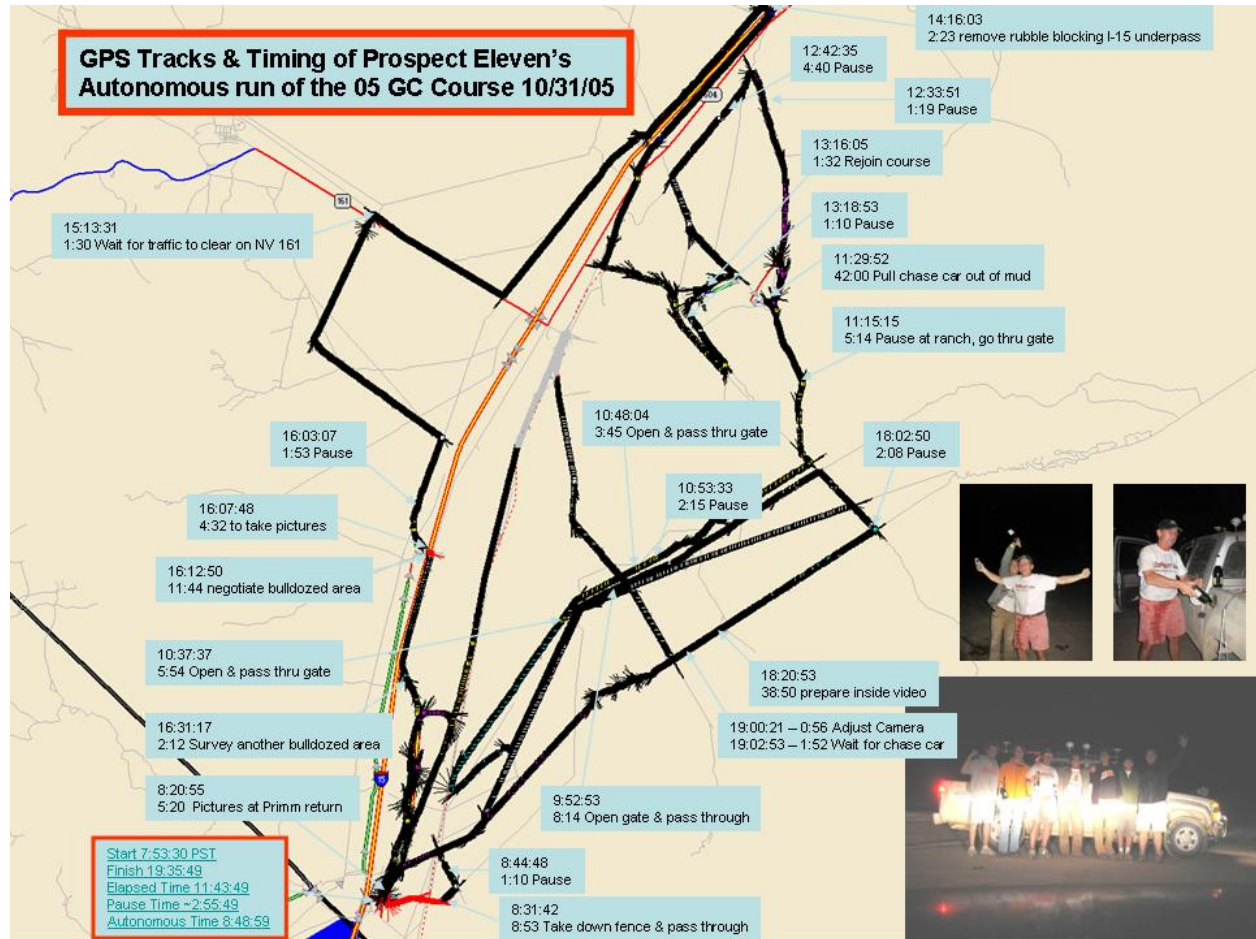
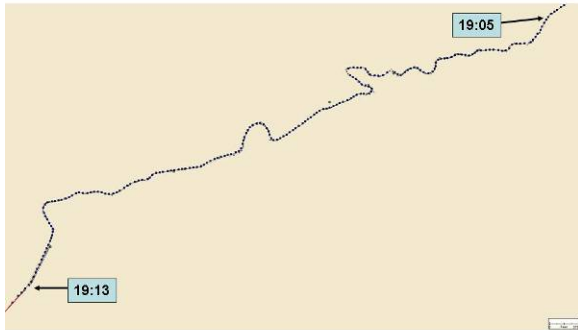


Figure 6.1 GPS tracks and timings of Post Grand Challenge run of 2005 Course

Except for the above constraints, none of which existed during the Grand Challenge, Prospect Eleven autonomously traversed the course. No changes, corrections or alterations were made to any of Prospect Eleven's autonomous systems. Prospect Eleven completed to course in 8 hours 48 minutes and 59 seconds of autonomous time, 11:43:49 of elapsed time. GPS tracks and timings for the run are presented in Figure 6.1. It can be argued that Prospect Eleven autonomously traversed an even more challenging course than that of the 2005 Grand Challenge. Except for the two lakes and the two "bulldozed" areas, Prospect Eleven ran autonomously, including places where the road was significantly rougher than what existed in early October.

What may be argued as being Prospect Eleven's highest achievement during the 2005 run was its 8-minute autonomous descent of Beer Bottle Pass at night beginning at 19:05. Only its headlights were illuminating the road ahead for its stereo camera to detected the edges and steer the vehicle down the treacherous terrain. It performed marvelously.

Night Time Descent of Beer Bottle Pass 19:13
2005 Grand Challenge Course



LAPRPA Grand Challenge, Unfenced Descents, October 31, 2005

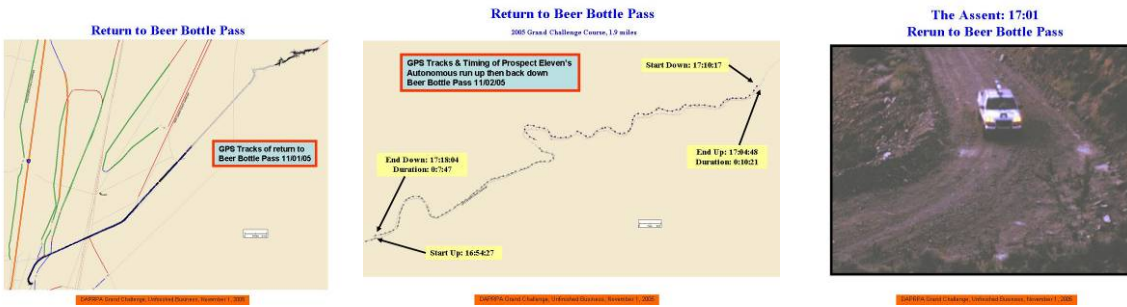
Beer Bottle Pass 19:10 PST
2005 Grand Challenge Course



LAPRPA Grand Challenge, Unfenced Descents, October 31, 2005

6.4.2 Return to Beer Bottle Pass

Because it was so dark the night before, we had no appreciation for what Prospect Eleven had accomplished during its final phases of its run of the 2005 course. We decided to go back to Beer Bottle Pass and rerun it in the daylight. Below are depicted the GPS tracks and an image of our autonomous return to Beer Bottle Pass on Tuesday, November 1. Prospect Eleven first ran up, then back down the Pass autonomously. Since we had gotten such a late start only the run up the pass was done in daylight (twilight), the run down had again to be done at night.



6.4.3 Assault on the 2004 Grand Challenge Course.

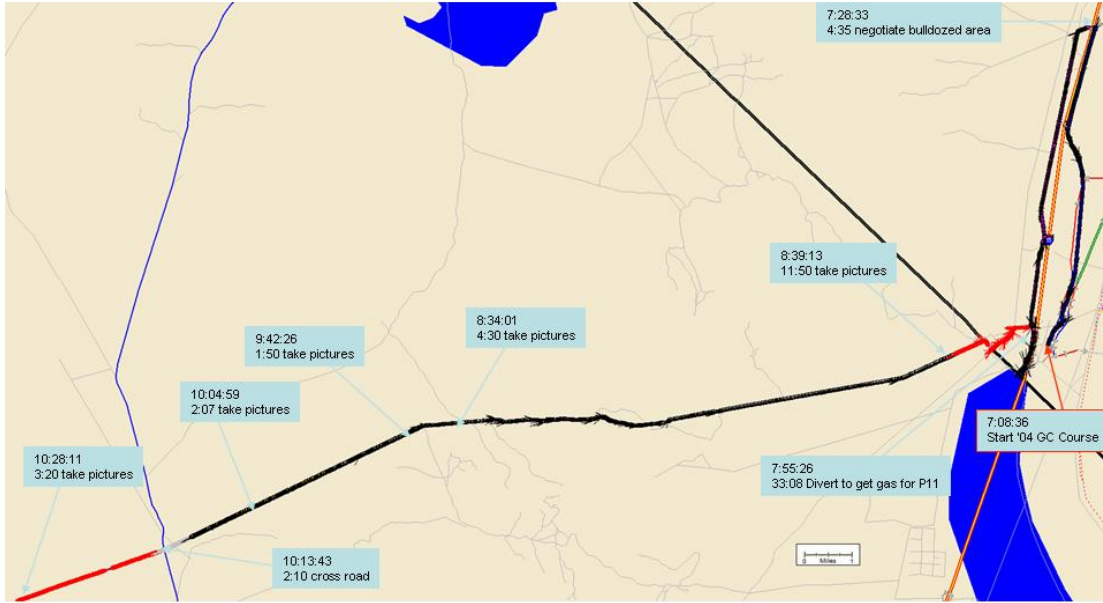
Given the success of the previous two days, it was decided to make an assault on the 2004 Grand Challenge Course on Wednesday, November 2. Since the vehicle was based in Primm and there was no knowledge on the state of the 2004 course, it was decided to make the attempt in reverse order, from Primm to Barstow by reversing the RDDF of the 2004 Grand Challenge Course and see how far we could go. Prospect Eleven was launched at 7:08:36PST and arrived at the Slash X Ranch at 18:00:00, for an elapsed time of 11h37m43s. It traversed the entire 2004 course in 8:09:00 of autonomous travel. While it did not encounter any lakes, manual control did need to be used to negotiate three places where the “road” had been completely washed put and was totally impassable by any vehicle and, at 17:32, to divert around an underpass that had been filled with silt such that there was less than 4 feet of clearance. Also, Prospect Eleven need to be repaired on two occasions. Once at when the steering wheel encoder froze at 13:06 and when the front left tire blew out at 17:16. Finally, it had to be pulled over to the side of the road on two occasions (16:33 and 18:41) to let traffic pass by. Otherwise, Prospect Eleven completed the entire 2004 course in 8:44:190 of autonomous travel. Most impressively, it autonomously



LAPRPA Grand Challenge, Unfenced Routes, November 2, 2005

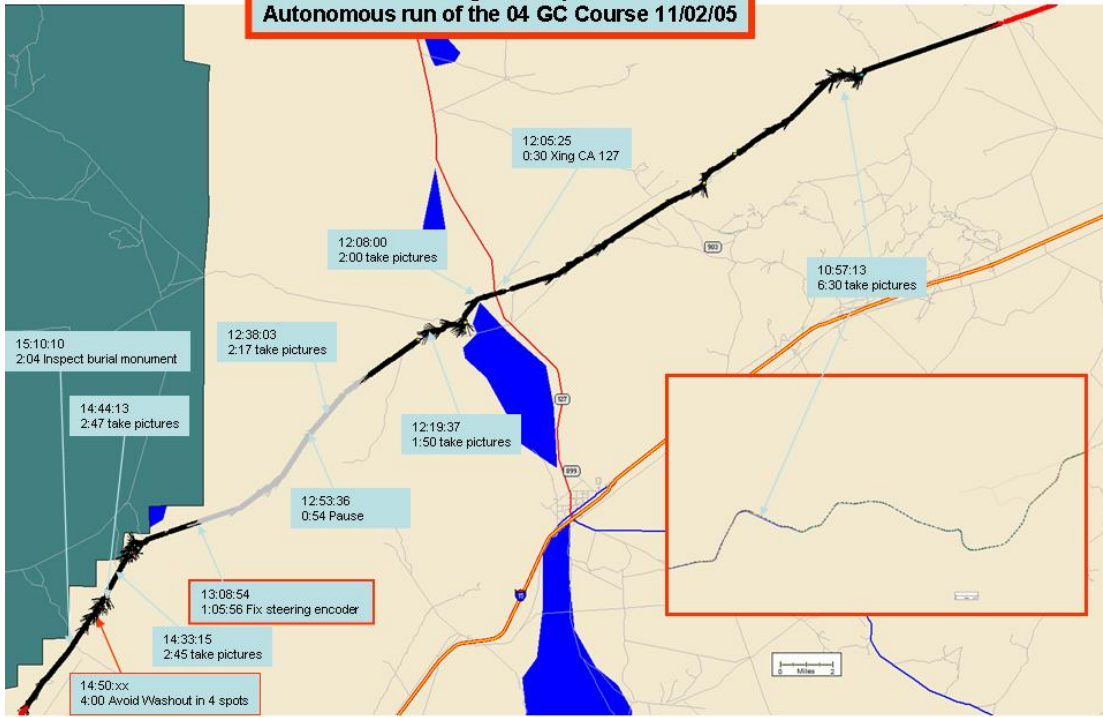
traversed Daggett Pass at night beginning at 18:18, just as it had negotiated Beer Bottle Pass on the previous days.

**GPS Tracks & Timing of Prospect Eleven's
Autonomous run of the 04 GC Course 11/02/05**



DAPRPA Grand Challenge, Unfinished Business, November 2, 2005

**GPS Tracks & Timing of Prospect Eleven's
Autonomous run of the 04 GC Course 11/02/05**



DAPRPA Grand Challenge, Unfinished Business, November 2, 2005

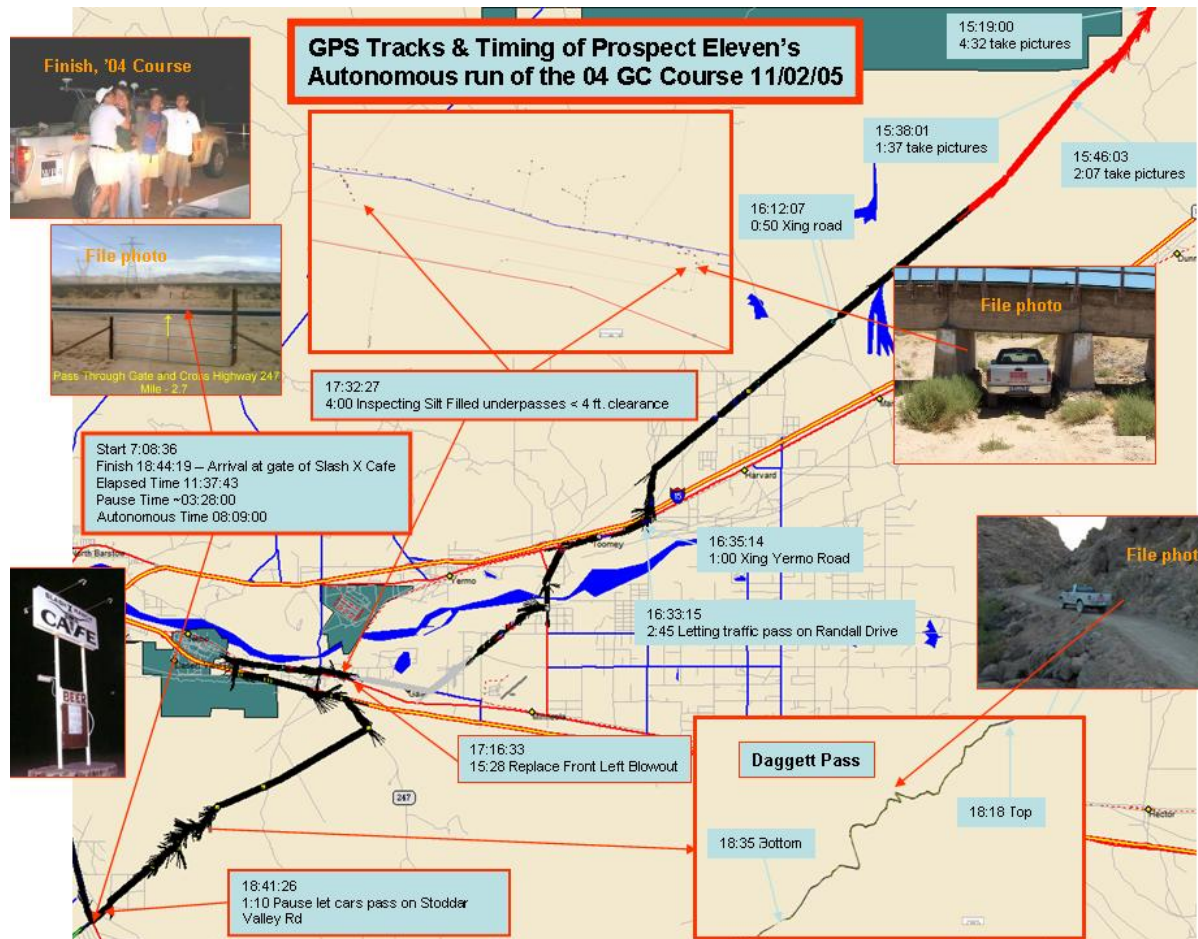


Figure 6.2 GPS tracks and timings of Post Grand Challenge run of 2004 Course

7. Final thoughts

Prospect Eleven turned out to be a phenomenal autonomous vehicle. It performed enormously better than we had any right to expect. The stereo vision system performed through a wide range of illuminations from bright daytime conditions through dark nighttime

8. References

1. A. Talukder, R. Manduchi, L. Matthies, A. Rankin, "Fast and Reliable Obstacle Detection and Segmentation for cross country navigation", in IEEE Intelligent Vehicles 2002.
2. W. van der Mark, F.C.A. Groen, and J.C. van den Heuvel, "Stereo based navigation in unstructured environments", in Proc. IEEE Instrumentation and Measurement Conference, 2001.
3. Alberto Broggi, Claudio Caraffi, Rean Isabella Fedriga, and Paolo Grisleri, "Obstacle Detection with Stereo Vision for off-road Vehicle Navigation", in Procs. Intl. IEEE Wks. on Machine Vision for Intelligent Vehicles, 2005.
4. Trucco, E. and Verri, A., *Introductory Techniques for 3-D Computer Vision*. Prentice Hall PTR, 1998.