Team Cornell:
Technical Review of the DARPA Urban Challenge Vehicle

Prof. Mark Campbell, Prof. Ephrahim Garcia, Prof. Dan Huttenlocher
Isaac Miller, Pete Moran, Aaron Nathan, Brian Schimpf, Noah Zych
Jason Catlin, Filip Chelarescu, Hikaru Fujishima, Frank-Robert Kline,
Sergei Lupashin, Max Reitmann, Adam Shapiro, Jason Wong

Cornell University
13 April 2007

Executive Summary

Team Cornell's 2007 DARPA Urban Challenge blends the team and technologies of a largely successful 2005 DARPA Grand Challenge with internationally recognized experts in probabilistic analysis, GPS, control systems and artificial intelligence to develop a vehicle capable of executing autonomous urban operations. Key state of the art technologies developed and integrated in Team Cornell's entry include: 1) reliable, high performance vehicle automation using brake, steering, transmission and throttle control; 2) tightly coupled vehicle position and attitude estimation with time synchronized antennas, square root information filtering, and integrity monitoring; 3) an integrated sensor suite of laser range finders, cameras, and radars to monitor the environment; 4) a novel probabilistic approach which fuses the many sensors in order to track moving vehicles relative to Cornell's vehicle; 5) a unique scene estimation algorithm, which condenses local probabilistic information to produce a lane-specific estimates of the environment used by all levels of planning; and 6) a hierarchical planning algorithm, with operational, tactical, and behavioral layers to achieve smooth local control despite switching between an expandable set of road behaviors and higher level mission goals. This document gives an overview of these key technologies, along with a sample of the experimental results.

DISCLAIMER: The information contained in this paper does not represent the official policies, either expressed or implied, of the Defense Advanced Research Projects Agency (DARPA) or the Department of Defense. DARPA does not guarantee the accuracy or reliability of the information in this paper.

Introduction

Team Cornell's DARPA Urban Challenge (DUC) program blends the team and technologies of a largely successful 2005 DARPA Grand Challenge (DGC) with internationally recognized experts in probabilistic analysis, GPS, and artificial intelligence to develop a vehicle capable of executing autonomous urban operations. Team Cornell has adopted a systematic design process during the short DUC development and testing cycle. Key elements of the development and test plan include: 1) systematic analysis and implementation using major development phases with a set of internal development milestones correlated with DUC milestones; 2) organization around six core technologies focused on the vehicle, sensing, estimation, and planning; and 3) development and testing at the Seneca Army Depot, a unique urban testing environment for the DUC.

Figure 1 shows important components of the development phases. Early in the design process, Team Cornell focused heavily on defining requirements to satisfy DUC scenarios,

which were then used to choose the sensor suite and its placement. Figure 1 (left) shows the earliest part of this process, where sensors were evaluated according to their ability to detect objects (cars, roads, etc.), and a 3D computer model was used to evaluate sensor coverage. After the initial sensor selection, a sensor vehicle was equipped with the suite of competition sensors, including laser range finders (LIDARs), radars, optical cameras, GPS, and a high precision inertial measurement unit. The sensor vehicle, used for data collection, algorithm development, and integration testing, is shown in Figure 1 (middle). The final phase in Team Cornell's development cycle is the validation phase using the competition vehicle. This phase includes the implementation of vehicle actuation, integration of hardware and software, and a long phase of testing, validation, and tuning the intelligent planner for different DUC scenarios. Figure 1 (right) shows Team Cornell's competition vehicle, a 2007 Chevrolet Tahoe.

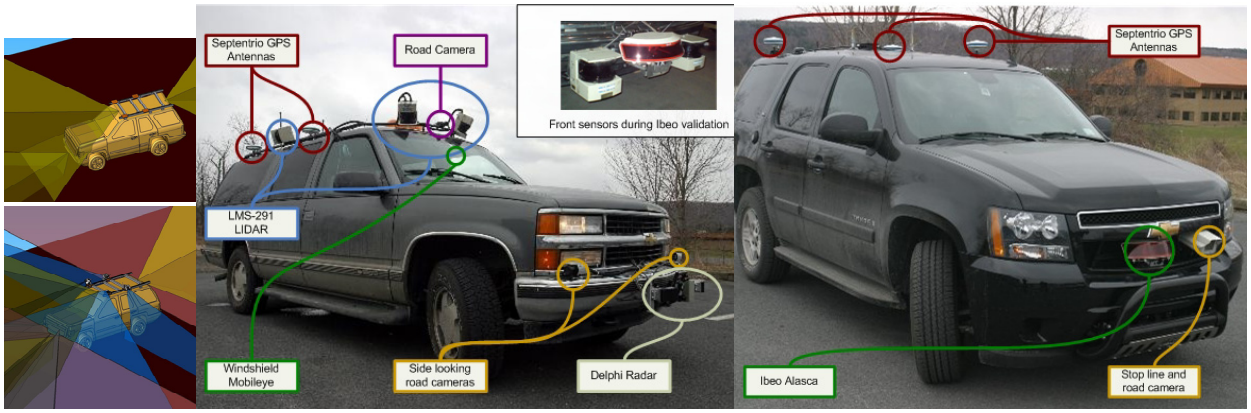

Figure 1: Progression of the design of the Team Cornell vehicle. Left: sensor analysis using 3D CAD models. Middle: Data collection with the sensor vehicle. Right: 2007 Chevrolet Tahoe competition vehicle.

Throughout this development cycle, Team Cornell has divided its technical approach along six major sub-problems identified in the DUC. These sub-problems, with their requirements, are:

Vehicle Preparation and Actuation: A stock SUV chassis is to be actuated and equipped for autonomous operation using standard aerospace motors. The solution must be robust during testing and in the competition, with low down-time and maintenance requirements.

Position, Velocity, and Attitude Determination: off-the-shelf inertial and navigation sensors are to be combined in a tightly-coupled position, velocity, and attitude (pose) estimator to facilitate localization in an Earth-fixed coordinate frame. The solution must be smooth and consistent despite GPS outages and signal distortion in the urban environment.

Obstacle and Environment Sensing: a diverse sensor suite is to be selected to sense all relevant aspects of the urban environment, including static and dynamic obstacles, ground plane, road lines, and curbs. The solution must be omni-directional and redundant.

Obstacle Identification and Tracking: the sensors are to be fused in a rigorous multi-target tracking scheme to generate a consistent map of static and dynamic obstacles near the vehicle. The solution must be able to recover from sensor mistakes, and must be independent of GPS.

Environment Structure Estimation: structural cues within the environment, such as stop lines, painted lane lines, and textural road boundaries, are to be combined with vehicle pose and tracked targets to create a final road-based map of obstacles with lane assignments. The solution must be probabilistically rigorous.

Intelligent Planning: A systematic process of determining vehicle actions based on the probabilistic model of the environment. The planner must accommodate asynchronous

operation, and must be stable and robust despite an unpredictable environment. The must also be expandable such that additional behaviors can easily be added as the system matures,
Figure 2 shows the architectural relationship between these technologies, which are described in detail within this document.

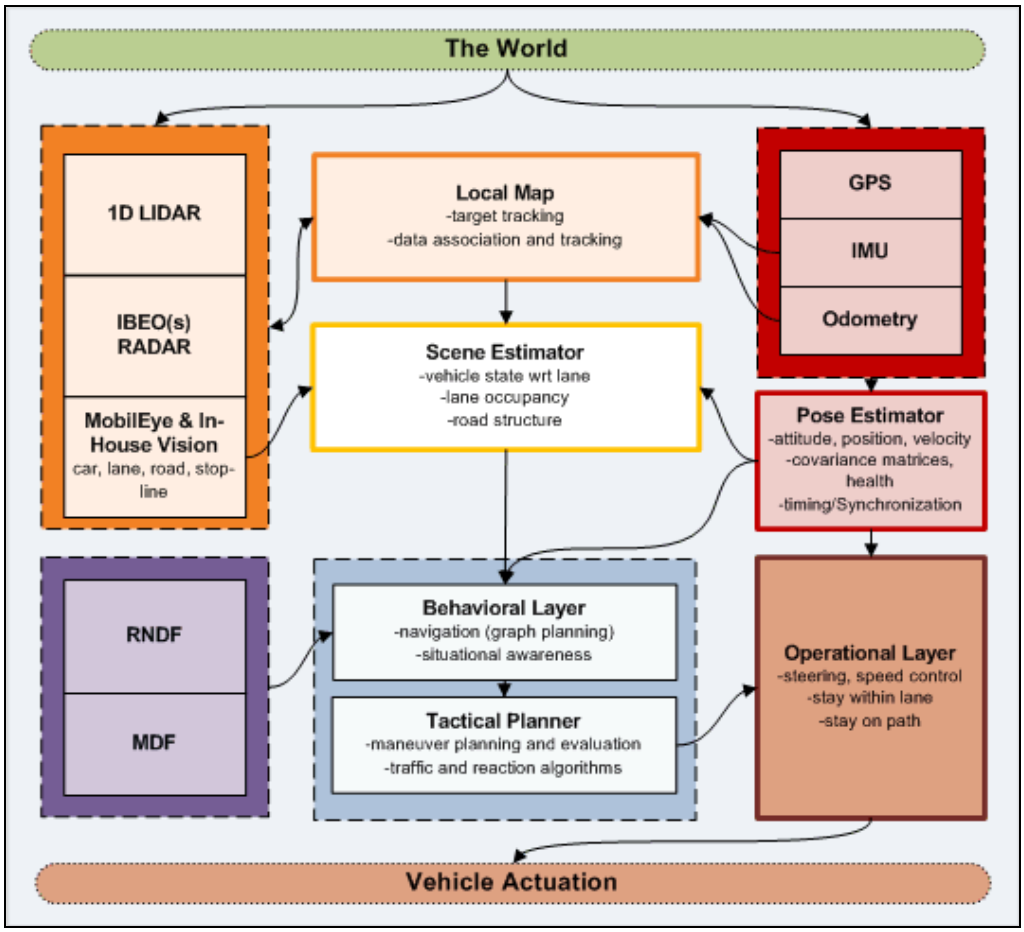

Figure 2: Architecture of the Team Cornell DUC entry.

A final key component of Team Cornell's development is a rigorous testing and validation procedure to ensure the functionality and robustness of all sub-components at all levels of integration. This final testing and validation occurs at the Seneca Army Depot in Romulus NY, which includes a large road network and vacated buildings. This simultaneous develop-and-test approach systematically integrates and tests new DUC scenarios with previously high-confidence components to ensure system maturity and robustness.

Vehicle Preparation and Actuation

Team Cornell's selection of the 2007 Chevy Tahoe and its subsequent conversion for autonomous operation were driven by two primary design requirements: responsiveness and reliability. The system must be quick to respond, without additional time delays and sluggishness beyond even a human's reflexes. The platform must be reliable because the DUC development cycle is too short to tolerate considerable down-time for vehicle repairs. Team Cornell addressed these two primary requirements with a design consisting of four components: the vehicle chassis, the power subsystem, the actuation, and packaging, each described below.

Chassis

Team Cornell's decision to adopt the 2007 Chevrolet Tahoe as its vehicle chassis was based on a number of carefully-considered design requirements intended to bolster responsiveness and reliability. First, the Tahoe is large enough to accommodate initial computer and hardware space requirements with extra room for additional computers as they are needed. Second, the additional size and weight make the Tahoe more likely to survive low-speed collisions without significant damage. Third, the stock Tahoe has a large engine bay with provisions for auxiliary power generation, whereas many smaller cars do not. Finally, the 2007 Tahoe comes equipped with an expansive set of easily-accessible onboard throttle, odometry, and health sensors, eliminating the need to modify the vehicle's electronics and throttle system.

The Tahoe also addressed many reliability issues that plagued Team Cornell's 2005 DGC entry. First, the manufacturer's warranty and stock parts ensure reliable operation and fast repair times. Second, the stock chassis has many rigid mounting points well inside the frame, allowing a computer rack to be mounted out of harm's way. Third, the 1776 lb. payload capacity ensures reliable operation despite a large number of computers. Finally, the common use of the Tahoe as an emergency vehicle makes a wide variety of after-market bumper, roll cage, and alternator kits available for additional reliability.

Power Subsystem

The design of Team Cornell's power subsystem was largely driven by lessons learned from the 2005 DGC. Based on that experience and a study of current computational hardware, initial power requirements were set at 2400 W, with an additional 1500 W budgeted for actuators at peak load. In addition, Team Cornell's power subsystem was designed to tolerate short periods with no power generation, due minor power and equipment failures which are inevitable. Finally, the system was designed to switch readily between onboard power generation and "shore power" in the laboratory, making the switch from field to laboratory testing simple.

Although Team Cornell used a separate power generator in the 2005 DGC, noise, heat, and reliability issues led the team to opt instead for a secondary alternator manufactured by Leece-Neville, paired with two Outback Power Systems inverters for the DUC. The secondary alternator functions independently from the stock electrical system on the Tahoe, so critical vehicle electronics do not fight with the computers for power. The Leece-Neville alternator provides 200 amps peak output at 24 volts, exceeding the design requirements, and the 3500 W inverters also exceed requirements with the capability of sourcing more than 6000 W instantaneously. In addition, the design can operate even if one alternator fails. Finally, the inverters generate clean enough power that sensitive electronics operate seamlessly, even during the transition from Tahoe to shore power.

The power subsystem also relies on four Optima deep-cycle automotive batteries for reserve power. These batteries are charged while the vehicle is being driven, and are able to provide peak power even when the vehicle idles. These batteries provide temporary power in cases when the vehicle must be restarted in the field.

Automation

The most significant decision affecting Team Cornell's development cycle was the choice to design and build the actuation scheme in house, for converting the Tahoe to drive-by-wire operation. This decision was only made after an extensive review of performance specifications, costs, and features of commercially available solutions, presented in Table 1. Three factors were considered critical in the decision. Scheduling was the first design constraint, and only the EMC

conversion and in-house actuation were determined to be feasible within the existing timeline. The second factor was the ability to automate two vehicles for redundancy; otherwise, the actuation would be a single point of failure throughout the development cycle. The final factor was cost, measured both in time and in money. Team Cornell's relationship with Moog Aerospace allowed the team to obtain actuators at no cost, and the knowledge and feasibility of repairing an in house system far outweighed the time spent designing the system.

Once Team Cornell decided to develop the vehicle actuation in house, a set of design specifications was created based on an analysis of the most demanding maneuvers the Tahoe might experience during the DUC. Steering and braking specifications were the most critical to vehicle responsiveness. A maximum steering angle performance requirement was defined as the ability to achieve a 700 degree steering wheel angle change in one second, which is required to produce a standard NHTSA fishhook maneuver to test commercial SUV rollover at 35 mph. A maximum braking force performance requirement was defined as the ability to achieve 100 lbf of pedal pressure within 0.1 seconds, which is a Class A stop in the Consumer Braking Information Initiative. Vehicle modifications must also be made without compromising the Tahoe's human interface and safety systems. As a result of this actuation design, the evasive capabilities of the Tahoe are limited only by the Tahoe itself, not by the physical limitations of its actuators.

Table 1: Summary of a trade study evaluating options for vehicle actuation.

	EMC	**AB Dynamics**	**Stahle**	**In-house**
Pros	- Complete integrated package - E-stop interface included - Powered from existing alternator - Minimal time investment for team	- Integrated brake, steering package - E-stop interface included - Powered from existing alternator - Digital interface - Well-documented - Installation in-house - Quickly transferable	- Digital interface - Well-documented - Installation in-house - Transferable	- Digital interface - Designed to specs - Installation in-house - Known architecture - Repairable
Cons	- Analog input - Performance specs unavailable - Must be installed by manufacturer at their facility - Nontransferable - Black box	- Requires additional transmission actuator	-Not complete package; steering actuator only -Requires 220 VAC	- Large time investment for team - Requires 120V AC power - Custom fabrication
Lead Time	5 weeks	4 months	3 months	2 months
Cost	$$	$$$$	$$	$

Packaging

Packaging the hardware within the vehicle created requirements ranging from usability to reliability to impact survival. The front, or passenger area, houses the actuation, actuation control, and allows for two passengers. The middle section, which replaces the middle seat, houses the inverters and battery bank. The low profile of these units reduces their center of gravity for safety, and allows team members access to the back of the computer rack. The rear of the vehicle houses the computer rack. The computer rack mounts rigidly to Tahoe frame while simultaneously addressing shock and vibration requirements that limit the lifetimes of sensitive and expensive electronics. Team Cornell opted for a custom, in-house design, with design flexibility, cost, and DGC experience being the primary factors. Team Cornell's custom built 1U form factor servers are highly optimized, with attributes such as a solid state hard drive for

reliability, identical components for redundancy, and dual core laptop processors for significant power and heat savings.

In addition to hardware location, the vehicle packaging controls cable flow. Roof mounted sensors and antennas interface through a waterproof (IP67) roof breach. Cables flow back to the top tray of the rack, and interface with their microcontroller counterparts. Ethernet cables flow down to the computing section. In contrast to the data, all power travels up from the low profile invertors and batteries.

Current Performance

Each component of the Tahoe – chassis, power, actuation, and packaging – has been implemented in the final competition vehicle, which has logged over 100 miles of driving (autonomous and manual) at the time of this writing. Figure 3 shows a block diagram of the implementation, which includes the four control components of the vehicle actuation, interface to the GM LAN, and interface to the intelligent planning architecture described later. Figure 4 shows three of these custom actuator systems built by Team Cornell in their final installation.

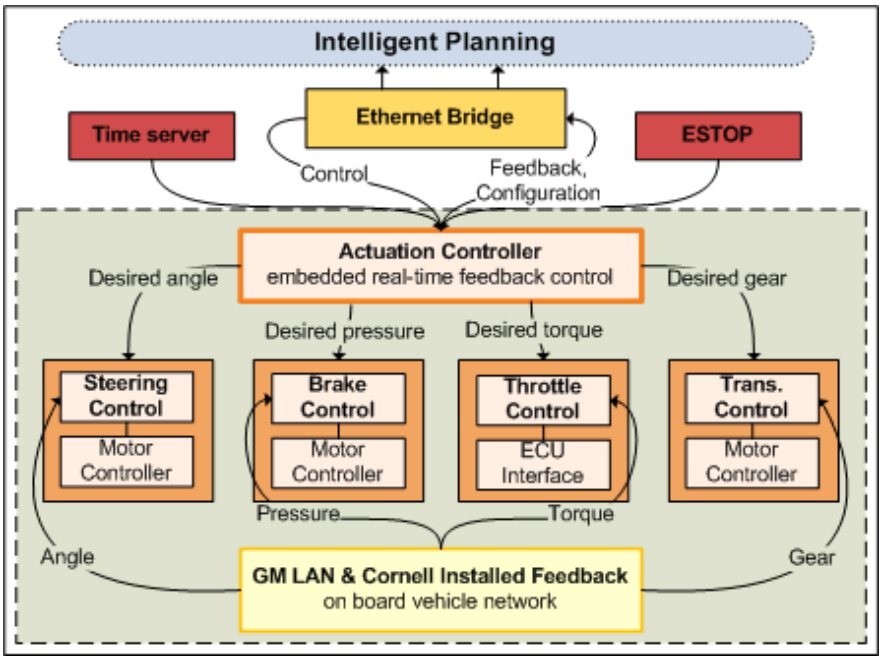

Figure 3: Implementation of the vehicle actuation hardware with operational control and intelligent planning.

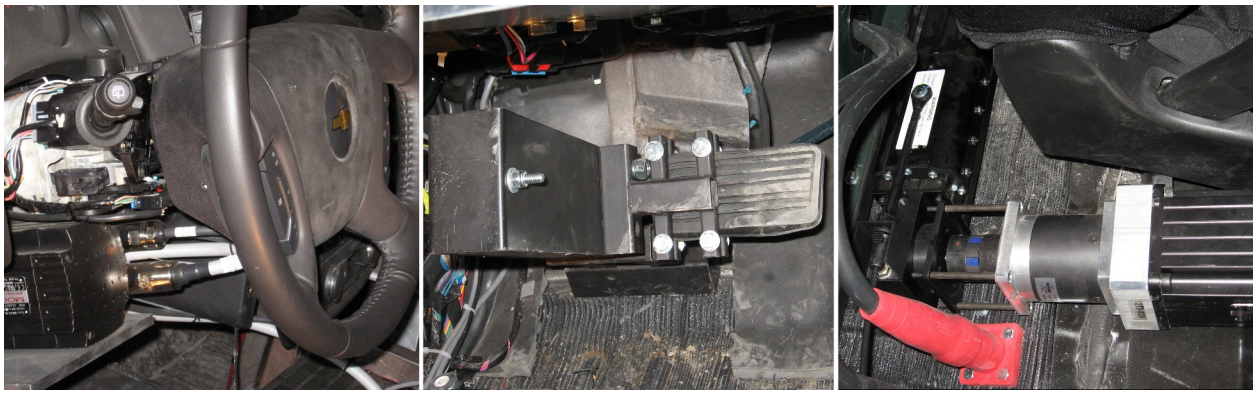

Figure 4: Team Cornell's custom-built actuators. Left: steering. Middle: brake. Right: transmission.

Position, Velocity, and Attitude Determination

In designing the Tahoe's position, velocity, and attitude estimation system (the pose estimator), Team Cornell considers three main objectives. First, the pose estimator must have in-plane accuracy at the meter level if the vehicle is to cross checkpoints, find parking spots, stop at stop lines, and navigate in environments where visual or other sensory cues might be weak or unavailable. Second, the pose estimator must be able to provide accurate estimates of differential vehicle motion, such as filtered angular rates and velocities, to allow external sensing systems to estimate various quantities in the dynamic environment from a vehicle-fixed coordinate frame. Finally, and perhaps most importantly, the pose estimator must be robust against the challenges of the urban environment. In particular, it must be able to withstand reasonable levels of satellite visibility and occlusion, as well as short blackouts and signal corruption in the urban canyon, without providing brittle or biased estimates of vehicle position.

These three design objectives have driven the development of Team Cornell's pose estimator. Most important in the early development process was Team Cornell's decision to design and build a pose estimator in house rather than buy an off-the-shelf alternative. In comparing these two options, the primary tradeoffs were time and risk spent developing our own system vs. money spent purchasing a top quality off-the-shelf alternative such as the Applanix POS LV system, which had already been proven successful in the DGC [1]. The POS LV is an attractive alternative because it is built precisely for the ground-based urban environment. It performs well because it fuses a tactical grade inertial measurement unit and vehicle wheel odometry measurements with high precision GPS in a tightly-coupled estimation algorithm. Team Cornell opted against it for several reasons. First, sponsorships from Northrop Grumman, Trimble, OmniSTAR, and Septentrio brought equivalent inertial and GPS sensors to Cornell at less than 5% of the cost of the Applanix. Second, Cornell has several faculty members with GPS and estimation expertise, making in-house design a viable alternative. Finally, the promise of closely monitoring the GPS signal environment for imperfections during testing and autonomous driving made it more desirable for Cornell to build its own pose estimator rather than rely on a commercial option. The following sections describe algorithm development, including design considerations meant to address the three primary objectives of the pose estimator.

The Pose Estimation Algorithm

Team Cornell's pose estimator fuses information from two sensors: a Litton LN-200 inertial measurement unit (IMU) and a Septentrio PolaRx2e@ GPS receiver. The LN-200 includes a three-axis fiber optic rate gyroscope and micro-machined accelerometers. The LN-200 measures inertial accelerations and rotations on the Tahoe, allowing the pose estimator to integrate the Tahoe's equations of motion in between GPS measurements and during GPS blackouts. The Septentrio is a three-antenna, single-clock GPS receiver that provides synchronized raw GPS measurements from visible satellites on all three antennas simultaneously. Team Cornell's pose estimator blends the two sensors in a tightly-coupled estimator, utilizing the IMU for fast update rates, differential vehicle motion, and data continuity, and utilizing the Septentrio for absolute positioning. The combination of these two sensors therefore addresses all three of the pose estimator's main design objectives: accurate absolute positioning, accurate differential vehicle motion, and a smooth, robust solution.

Cornell's pose estimator is implemented as a Square Root Information Filter (SRIF), a numerically robust implementation of the Extended Kalman Filter (EKF) [2]. The SRIF performs approximate Bayesian estimation in two steps, prediction steps and update steps. The prediction

step is equivalent to dead reckoning, whereby the vehicle equations of motion are numerically integrated forward with IMU measurements. Each dead reckoning step is a discrete Euler step accounting for external disturbances such as gravity, the rotation of the Earth, and centripetal and coriolis accelerations [3],[4]. A numerically stable Legendre polynomial construction algorithm is used within this integration to implement the full 360x360 EGM-96 gravity potential model for the most accurate gravity subtraction [5]. Additionally, the Septentrio receiver clock offset and clock drift rate are also integrated using a second order random walk model [6].

In the update step, raw GPS measurements generated by the Septentrio are fused with the current predicted pose estimate to create a blended posterior pose estimate. Most important in this step is the fact that the solution utilizes only the raw GPS measurements in the SRIF, and does not use the processed position, velocity, and attitude solution generated by the Septentrio. The pose estimator, therefore, is a tightly-coupled navigation solution with added robustness over a black box GPS receiver. This strategy is similar to the Applanix POS LV solution, except that Team Cornell does not (at present) utilize the OmniSTAR service. Trust placed in OmniSTAR was one of the causes of Team Cornell's crash in the 2005 DGC, and Team Cornell sought to avoid similar problems except when centimeter level positioning was required [7].

The GPS measurements in the pose estimator are raw pseudorange measurements and double differenced carrier phase measurements. High-fidelity experimentally-proven measurement error models are utilized for both pseudorange and double differences, accounting for placement of the antenna triad with respect to the IMU, clock errors, ionospheric delays, GPS ephemeris errors, and relativistic errors [8]. In addition, an accurate tropospheric delay model is included to account for signal delay and distortion due to moisture and gas density in Earth's lower atmosphere [9],[10]. Each pseudorange is also augmented with a time-correlated residual line of sight bias from the source satellite to account for the effects of multipath in the urban environment. These residual biases help absorb unanticipated signal delays due to the presence of buildings and foliage while keeping the estimator accurate and statistically significant. The pseudoranges and double differences are modeled with noise dependent on tracking carrier to noise ratio and correlations across the three Septentrio antennas to preserve filter integrity [11].

To ensure robustness within the pose estimator, the raw GPS measurements are carefully scrutinized before being incorporated into the posterior pose estimate. First, the full SRIF update is performed with all raw GPS measurements. Before broadcasting this posterior estimate, however, a chi-squared hypothesis test is performed to ensure the GPS measurements match the pose estimate at a statistically significant level [6]. If the pose estimate fails the hypothesis test at the 99% significance level, the individual measurements are tested with chi-squared statistics one by one. These individual integrity monitoring tests are used to reject measurements from individual satellites without throwing out the entire batch of GPS measurements, allowing the pose estimator to incorporate as many valid measurements as possible, even in harsh signal environments. In addition, the measurements are subjected to a final chi-squared hypothesis test to check whether the entire batch of measurements should be thrown out even after distorted measurements have been removed. This software integrity monitoring augments the Septentrio's multipath rejection technology to keep the pose estimate accurate despite the urban environment.

Team Cornell's pose estimator reports accurate estimates of vehicle yaw, pitch, roll, Earth-fixed position and velocity, rate gyro biases, accelerometer biases, clock offset, clock drift rate, residual satellite biases, and double difference ambiguities at 100Hz. The pose estimator is implemented in C++ on a dual-core Pentium-Mobile processor running Windows Server 2003. Integration steps run at 400 Hz, full prediction steps at 200 Hz, and GPS updates at 1 Hz.

Pose Estimator Performance

The pose estimator has seen many hours of testing and validation on both the sensor testing Suburban and competition Tahoe. The discussion below describes one of these tests: the results of a typical hour-long drive around the Ithaca area. The vehicle's path during this test is highlighted in blue in Figure 5 using Microsoft's Virtual Earth software. Of particular interest is Figure 5 (right), which shows some of the largest absolute positioning errors experienced during this test (approximately four meters, one lane width). Note that the vehicle traveled over the same ground position more than once during this test, demonstrating the repeatability of the pose estimator. In addition, the pose estimator sustained repeated satellite occlusions, signal drops, and distortions due to multipath without resulting in significant increases in localization error. The estimator also withstood several small (less than ten seconds) GPS blackouts during this run. Figure 6 shows the pose estimator's ground track during a three point turn in a difficult GPS signal environment. Once again, the estimator proves to be consistent and robust.

The performance shown in Figure 5 and Figure 6 is achieved using only the LN-200 IMU and the Septentrio GPS antenna receiving the civilian L1 GPS signals. Based on lessons from the 2005 DGC, this architecture was tested first in order to prevent masking of problems using wheel odometry, WAAS differential corrections, and OmniSTAR high precision signals. Both WAAS and wheel odometry have been integrated and tested, resulting in better performance. In particular, WAAS corrections improve absolute positioning of the civilian L1 signal in even the harshest multipath environments. Vehicle wheel odometry effectively smooths the estimates and extends the estimator's capability to withstand minute or longer blackouts, as found in the DGC [8]. Finally, the OmniSTAR high precision signal can be carefully incorporated, with restrictive hypothesis tests, when absolute positioning at the decimeter level is critical to mission success. This will enable the vehicle to find parking spots and stop lines in the absence of visual cues.

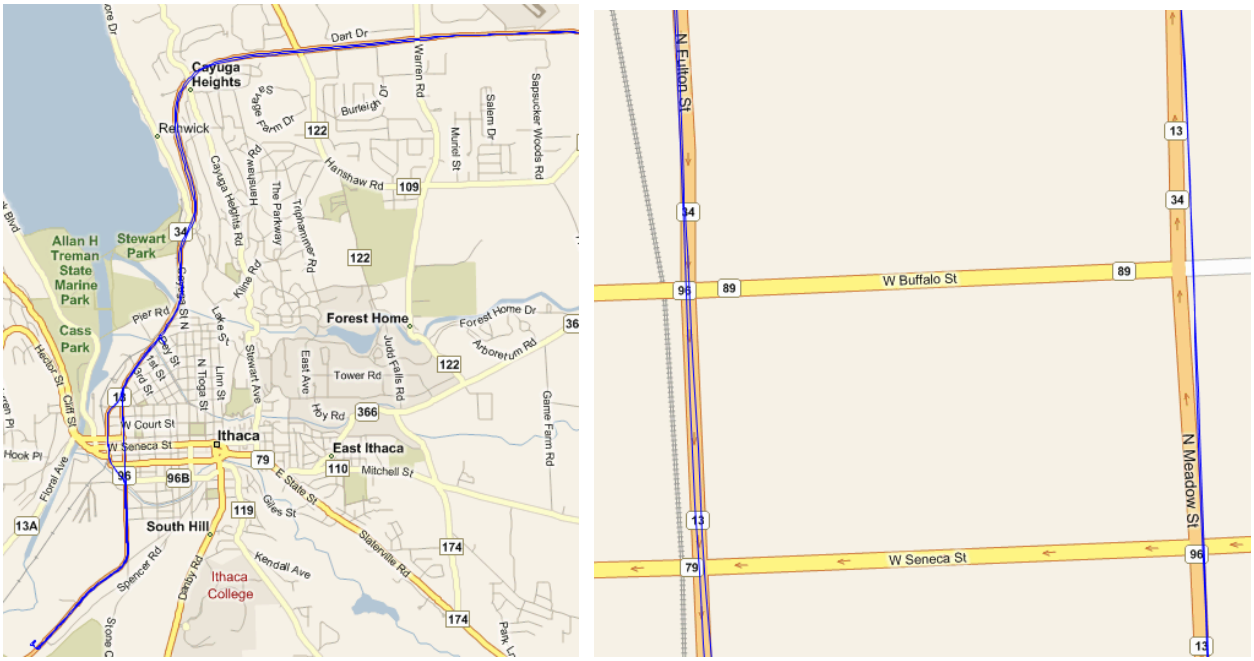

Figure 5: Left: the Cornell pose estimator running as the test vehicle drives through downtown Ithaca. Right: the pose estimator localizes the vehicle with up to one lane width of absolute positioning error.

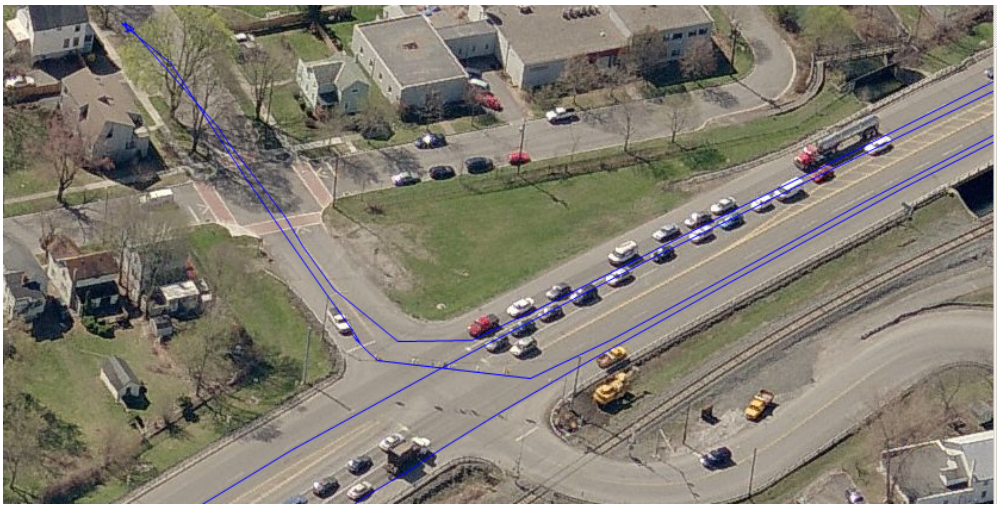

Figure 6: Pose estimator ground track during a three point turn in a difficult signal environment

Obstacle and Environment Sensing

The Tahoe's sensing system is designed around the challenging requirement of being prepared to see everything in an unknown urban environment. This broad requirement is further subdivided into the ability to detect three types of environmental features: static obstacles, moving obstacles, and various types of roads. Most importantly, Team Cornell has evaluated and selected its sensors based on their capability of *detecting* these three aspects of the environment, not on their ability to *track* these objects independently. In this way Team Cornell adopts a purely Bayesian approach to sensor fusion, preferring to fuse raw sensor output in a centralized estimation scheme rather than depend on individual sensors' proprietary tracking algorithms. The current sensor suite is shown in Figure 7, and includes two 1.5D LIDARs, four 1D LIDARs, three millimeter-wave radars, and several black and white cameras.

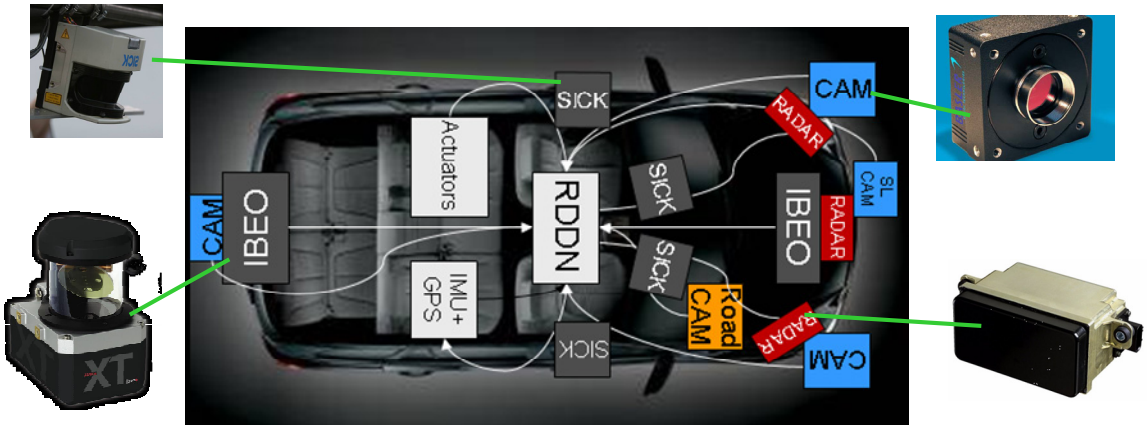

Figure 7: Sensor layout of the Team Cornell DUC vehicle.

Sensor Suite Design

The high cost of most sensors and their importance to the sensor fusion process forced Team Cornell to perform a careful and systematic analysis of their strengths and weaknesses

against anticipated DUC scenarios. This analysis initially considered sensor coverage, where sensors were considered for their detection capabilities in terms of azimuth coverage. At the same time, sensors were also considered for their coverage capabilities out of the plane, ensuring object detection even under vehicle pitch and roll induced in a typical urban environment. Figure 8 shows top-down and isometric views that were used in this study. The second phase of the sensor evaluation process involved researching detection capability of each sensor in critical DUC scenarios, shown in Table 2. This process ensured the final sensor suite would be able to detect all necessary objects in the most critical DUC scenarios with as much redundancy as possible. The sensor fusion algorithms then appropriately consider the many sensors in building a probabilistic model of the environment.

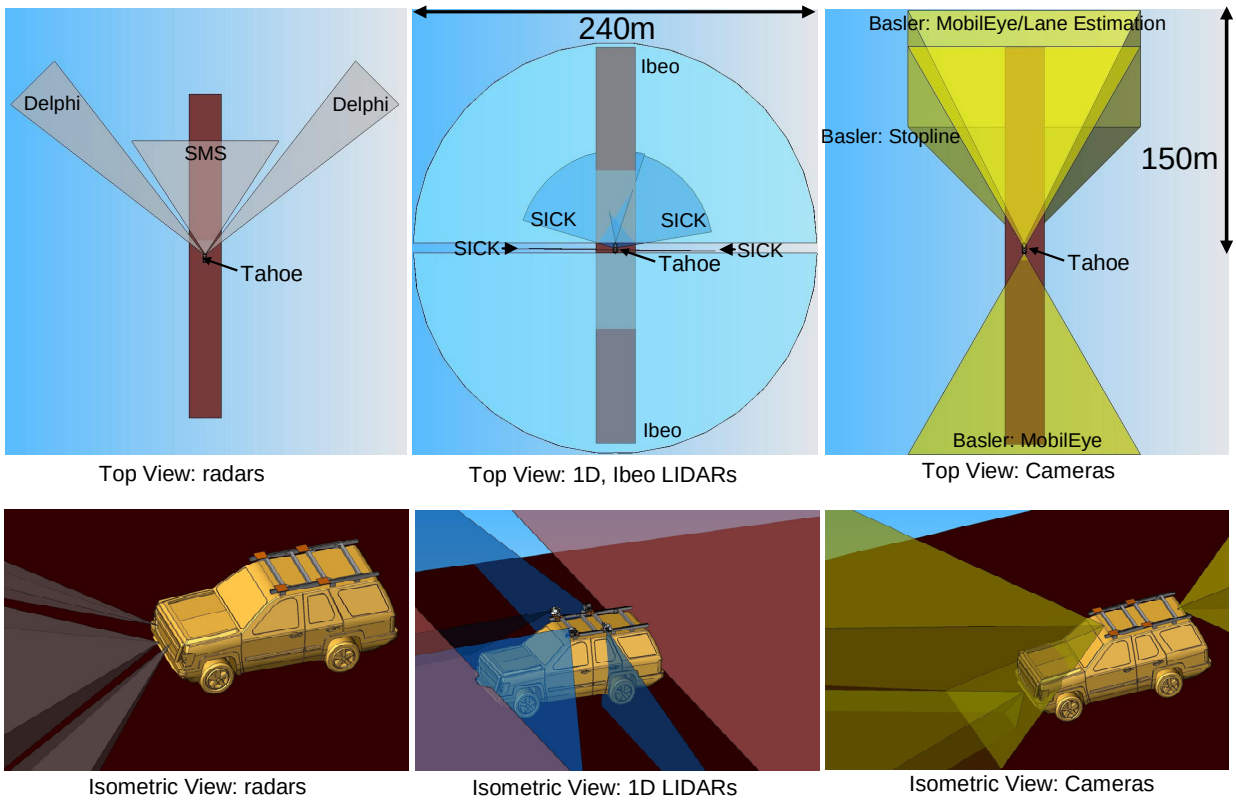

Figure 8: Fields of view for each sensor type, in top down and isometric views.

The final sensor suite selected from the design analysis includes both active and passive electro-optical sensors. Active sensors include two Ibeo AlascaXT 1.5D LIDARs, four SICK LMS291 LIDARs, two Delphi 76 GHz FLR radars (+/-7.5 deg field of view, 150m range), and an SMS UMRR Mid-Range radar (+/-35 deg field of view, 70m range). Passive sensors include five Basler machine vision CCD and CMOS cameras. Each sensor is chosen for a particular detection capability. The Ibeos are Team Cornell's primary sensors for detecting vehicles and static obstacles, as they each scan with four nearly-parallel beams out to 240 m range, with 180° fields of view. Furthermore, the internal geometry of these beams allows the Ibeos to be used to separate ground returns from obstacle returns reliably. Alternatives to the Ibeos, including standard single-scan LIDARs and the Velodyne 2D LIDAR, were studied extensively. The 1D LIDARs, while relatively inexpensive, are bulky and tend to miss cars. The Velodyne is much

more expensive and provides a complete 3D point cloud, but Team Cornell decided the additional data provided little information once a cluster of points was determined to be an obstacle. The Ibeos are therefore chosen as a compromise both in price and tolerance to vehicle pitch and roll. The three radar units complement the Ibeos in their ability to detect moving objects using Doppler shift, and are thus mounted to see oncoming traffic and to scan intersections from the front of the vehicle, allowing rapid target acquisition and situation comprehension. Visual cameras using MobilEye SeeQ software are also used to augment the Ibeos and radars in detecting moving vehicles from the front and the rear of the vehicle.

Table 2: Sensor capabilities mapped against typical DUC scenarios

	Front Corner Cameras	Front Camera	Stop-line Camera	Front Ibeo LIDAR	Front Corner radars	Front Center radar	Front SICK LIDAR	Side SICK LIDAR	Rear Camera	Rear Ibeo LIDAR
Staying in Lane		✓					✓	✓	✓	
Stop <1m from Stop Line		✓	✓							
Maintain Vehicle Separation		✓		✓		✓				
Vehicles at an Intersection	✓	✓		✓	✓	✓				
Leaving Lane to Pass		✓		✓	✓	✓		✓		
Return to Lane after Pass		✓		✓	✓	✓		✓		
U-Turn	✓	✓		✓	✓	✓	✓	✓	✓	✓
Following a Vehicle		✓		✓		✓				
Queuing at an Intersection	✓	✓		✓	✓	✓				
Negotiate Obstacle Field		✓		✓						
Road Blockages				✓			✓	✓		
Merging to Traffic Circle	✓	✓		✓	✓	✓	✓	✓	✓	✓
Sparse Waypoints (straight)		✓		✓			✓	✓		
Road Follow: GPS Outages		✓					✓	✓		
Merging: T Intersection	✓	✓		✓	✓	✓				
Merging: 4-way Intersection	✓	✓		✓	✓	✓				
Left Turns at Intersections	✓	✓		✓	✓	✓				
Emergency Vehicle Avoid				✓			✓	✓		✓
Blocked Intersection				✓						

Visual cameras and 1D LIDARs combine to form Team Cornell's road detection system, with an ultimate goal of providing measurements of the Tahoe's heading with respect to the road and distance from the center of the road for estimating the structure of the environment. The system also determines the number of lanes and a piecewise linear road parameterization using graph-based segmentation [12]. MobilEye's lane detection software augments Team Cornell's texture-based segmentation algorithm. This commercially-available software uses detected lane lines to determine the Tahoe's orientation with respect to the road as well as to estimate a road parameterization. These two visual systems are augmented with side-facing 1D LIDARs to detect curbs and rough textures, as well as two forward-facing 1D LIDARs to help estimate the ground plane near the Tahoe. Finally, Team Cornell's road detection system is completed with a downward-facing camera running a stop line detection algorithm based on the Hough transform and Canny edge detection.

Sensor Networking

To avoid using proprietary hardware to interface to each individual sensor and to improve the modularity of the sensor system, Team Cornell uses dedicated real time microprocessors to

interpret, timestamp, and broadcast sensor data over a standardized UDP Ethernet network. This Real-time Data Distribution Network (RDDN) is composed of a 100-BaseT network of Motorola 9S12NE64 microcontrollers, each with embedded Ethernet MAC and PHY. This RDDN allows sensor data to receive accurate time stamps through synchronization with a master microcontroller, a benefit critical to higher level sensor fusion. The common Ethernet interface also allows any computer to listen to any sensor without specialized hardware, and it allows real-time data to be simulated in playback over the network.

Current Performance

Interfaces to every sensor are completed and tested in their final configuration, including integration into the RDDN. This system achieves less than 1ms timing error on each sensor on the Tahoe. Ibeo data and radar data are currently streamed directly to the obstacle mapping computer for real-time processing. Team Cornell's road tracking system currently runs in real-time on a dual processor computer. Stop Line detection also works in real-time and can detect a stop line up to 10 m from the front of the vehicle. Examples of vehicle, road, and stop line detection in Team Cornell's road tracking system are shown in Figure 9.

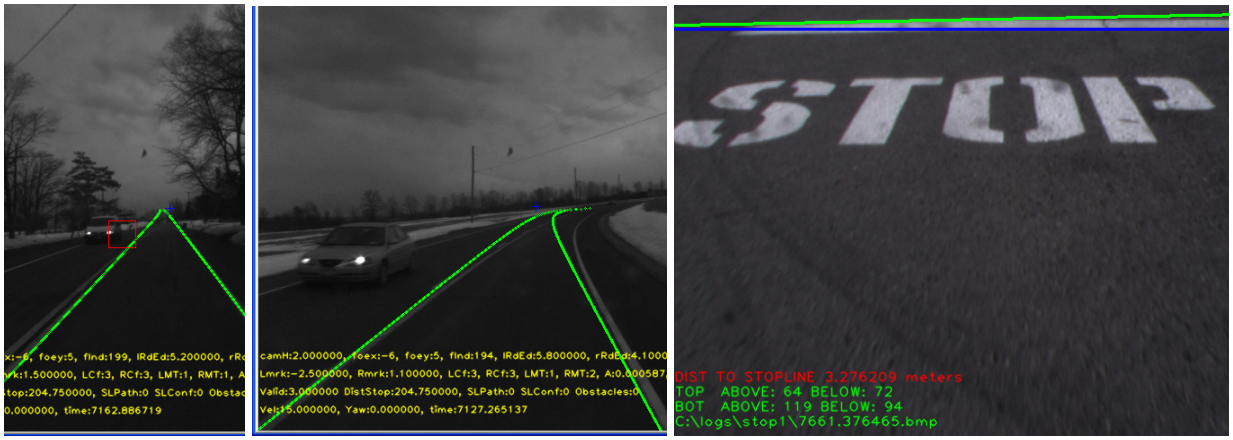

Figure 9: Real time vision system implementation using MobilEye software, cameras, and Cornell segmentation software. Left: vehicle detection. Middle: lane detection. Right: stop line detection.

Obstacle Identification and Tracking

The purpose of Team Cornell's obstacle identification and tracking algorithm, called the Local Map, is to fuse the output of all sensors that detect obstacles in the unstructured environment around the vehicle. Team Cornell derived and implemented the Local Map in response to four primary needs. First, autonomous driving mandates at least rudimentary sensing and obstacle detection in order to identify safe and traversable portions of the environment. In this regard, the Urban Challenge is significantly more challenging than the two previous Grand Challenges, which assured that there would be no interactions among moving entities. Typical fixed-size approaches such as occupancy grids and terrain maps are not appropriate for the task. Second, the model must be descriptive and have predictive capability; that is, the need for more information than mere location of each obstacle. This objective stems from the assumption that effective path planning higher requires the ability to predict the locations of obstacles forward for short periods of time. As that prediction's accuracy increases, the vehicle is better able to anticipate the locations and even the actions of obstacles it may need to avoid in the future.

Third, the Local Map must be consistent. Each sensor may potentially give conflicting interpretations of the environment near the vehicle; Local Map must resolve sensing ambiguities to present a coherent interpretation of the surroundings to the path planner. The consistency objective also implies consistency across time, as large changes in the interpretation of the sensor data tend to lead to indecision in the path planner. Finally, the Local Map must be robust. Because the populated urban environment changes constantly, the number of visible obstacles and the maneuvers they perform change rapidly. The Local Map must be able to track these changes without instabilities from uncertain sensors.

Team Cornell's Local Map algorithm specifically addresses each of these four issues. A decision was made early on to opt for a centralized estimation and sensor fusion scheme, so that each sensor would supply measurements to be fused into the Local Map, in order to produce a single consistent interpretation of the world. By adopting centralized Bayesian estimation techniques, each sensor could be incorporated with all other sensors into a consistent world interpretation. The robustness of the centralized Bayesian view of the world became much clearer as Team Cornell began to sample off-the-shelf sensors that included their own proprietary tracking algorithms. In particular, most off-the-shelf tracking algorithms proved to be quite brittle in practice, either by making strong assumptions about vehicle motion or simply due to the inability of a single sensor to estimate the entire urban environment. By combining the raw information from multiple sensors, each with different strengths, the centralized Local Map had the best chance of providing the information the path planner required.

After settling on a centralized estimation scheme, Team Cornell began to evaluate existing options in the target tracking literature. The EKF seemed a natural choice for tracking both stationary and moving targets, and early experiments with a stationary LIDAR and moving obstacles, showed it was possible to estimate a number of higher order parameters of target motion, including speed, heading, curvature, and even length and width.

While the EKF successfully tracked static and moving obstacles, a key limitation is that it required knowledge of which measurements corresponded to each obstacle. That knowledge is not available in arbitrary urban environments, where even the overall number of obstacles is unknown. A number of target tracking algorithms exist to handle this situation, including the Joint Probabilistic Data Association Filter (JPDAF), the Multiple Hypothesis Tracking algorithm (MHT), and several variants of Monte Carlo Data Association (MCDA) [13], [14]. Of these approaches, the traditional JPDAF and MHT seemed better suited for small numbers of targets due to computational expense. So instead, a novel approach to using a modified MCDA-based Rao-Blackwellized Particle Filter (RBPF) Team Cornell's Local Map was adopted [15].

The RBPF Local Map Algorithm

Team Cornell's custom RBPF algorithm begins with the assumption that both the data associations $N_{1:k}$ *and* obstacle states $X_{1:k}$ must be estimated in order to accurately represent a dynamic environment, where the number of visible obstacles changes rapidly in time. Both of these variables could be estimated with a large particle filter, but the high dimensionality of the resulting state vector would mandate an infeasible number of particles to approximate the posterior distribution $p(N_{1:k}, X_{1:k} | Z_{1:k})$ conditioned on a history of raw sensor measurements $Z_{1:k}$. Instead, the posterior distribution can be factorized using conditional probability to yield:

$$p(N_{1:k}, X_{1:k} | Z_{1:k}) = p(N_{1:k} | Z_{1:k}) \cdot p(X_{1:k} | N_{1:k}, Z_{1:k})$$

The second term, $p(X_{1:k}|N_{1:k}, Z_{1:k})$, is simply the estimation of obstacle states when the measurements and the data associations are known, which is a standard tracking problem solvable by a traditional EKF. The problem of estimating the data associations, $p(N_{1:k}|Z_{1:k})$, can be solved by a small particle filter. In this way, the massive nonlinear, non-Gaussian dynamic environment estimation problem reduces to a series of decoupled obstacle tracking problems governed by data association decisions made by a small particle filter. Intuitively, each particle within the particle filter makes its own data association decisions and therefore develops its own list of obstacles and estimates of obstacle motion. The dynamic obstacle environment is then approximated by the set of all obstacles in all particles, and the path planner can act either on the information provided by the most likely particle, or on the weighted information aggregated from all particles.

The Local Map algorithm is successfully fast and efficient because the EKFs estimating each obstacle are rich enough to distinguish obstacles from one another except in short-lived cases where one obstacle passes close to another. In other words, the data associations are either "obvious" or limited to a few likely choices, so a very small set of particles is adequate to represent all the most likely data association decisions to be made near the Tahoe. The small set of particles provide more robustness than a deterministic data association algorithm, as particles that have made more data association mistakes are likely to be sampled out of the RBPF in favor of particles that more accurately match the dynamic environment [15].

Most importantly, the Local Map makes no explicit attempt to separate static and dynamic obstacles or treat them differently. Instead, all obstacles are tracked with estimates of position, heading, speed, curvature, and size, so that no hard decision is ever made about whether the obstacles are static or dynamic. This general tracking model within the Local Map is an excellent representation of the urban environment, where parked cars may begin to move, moving cars may temporarily come to a stop, and there are no sensors that are able to distinguish easily between cars and non-cars. The algorithm also does not depend on any ad hoc track initialization. Because data associations to new obstacles are permitted in addition to associations to existing obstacles, each particle within the Local Map automatically determines the number of targets it wishes to track. This allows the Local Map to create new obstacles as they come into view, and to drop old obstacles when they are out of sensing range.

Results and Performance

Figure 10 shows the output of the most likely particle in the Local Map during a typical sensors test run on Team Cornell's Suburban. Here, Ibeo laser data is combined with the sensing vehicle speeds and rates of rotation to track all car-sized obstacles as potentially moving targets. In this particular frame, two moving cars have been identified from historical data, as well as eight car-sized obstacles within sensor range. In this configuration, the Local Map is generally able to track obstacles with less than 0.5m position error out beyond 100 m range, with the Ibeo returning sensor hits up to 300 m away. The algorithm has successfully mapped urban environments with more than 50 identified obstacles. It is important to note that these results use Ibeo tracking data only; as radar and vision data are added, performance will improve. Also, the Local Map is computed in a vehicle-centric coordinate frame and makes no assumptions about road structure. It is therefore independent of GPS, so the Tahoe can identify and track obstacles even if its absolute position is unknown.

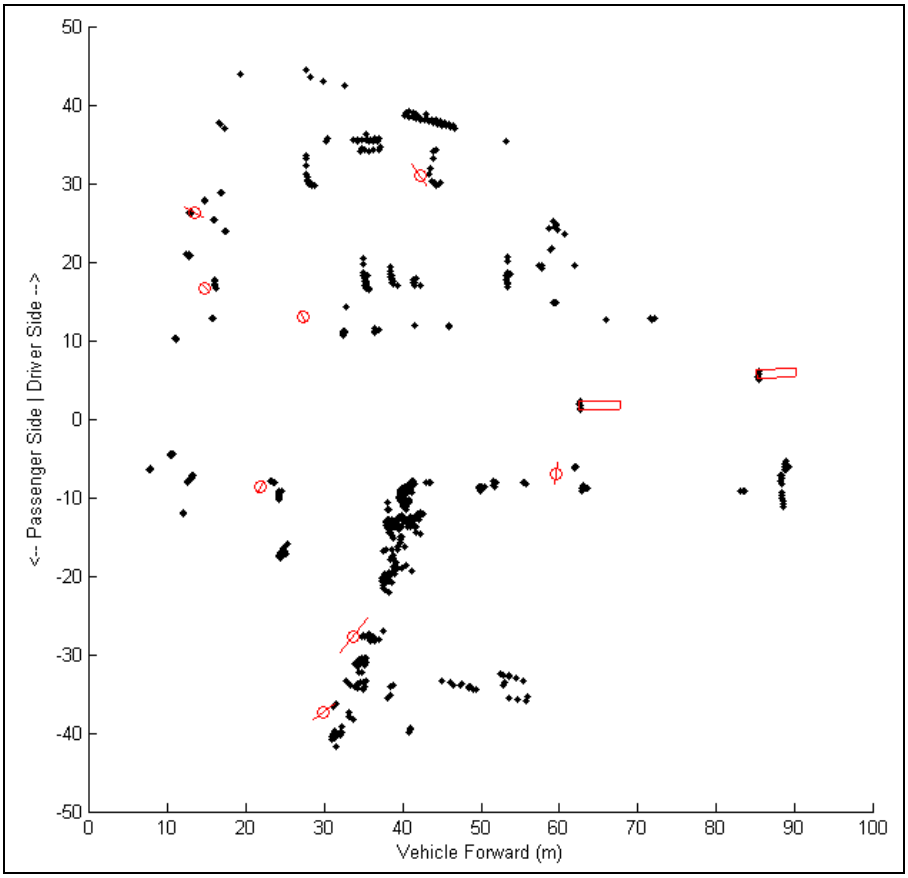

Figure 10: Output of the Local Map from a moving vehicle in a typical urban environment. Two moving cars (red boxes) have been identified, as well as several other car-sized obstacles (red circles). Raw LIDAR hits are shown as black points. The ego-vehicle is located at the origin toward the x-axis.

The Local Map algorithm estimates obstacle lists in the environment surrounding the vehicle at 100 Hz on a dual-core Intel machine running Windows Server 2003. A total of five particles are used to represent Local Map's data association decisions, though there is significant room for expansion. Both the prediction and the update steps within each particle may be parallelized, so that each particle is processed separately in its own thread. On a dedicated dual-core machine, such an optimization will cut computation time nearly in half. This generates the possibility of adding richer and more descriptive obstacle EKFs as the DUC testing matures.

Environment Structure Estimation

Team Cornell's Local Map provides a list of obstacle states and measures of confidence to the path planner at 100 Hz, which, in the worst circumstances, can be used to avoid collisions safely. The Local Map intentionally does not take into account the constraints typical of normal driving, such as the road boundaries and structure, as other drivers on the road may not always follow these constraints. In normal driving scenarios, however, when all agents on the road obey the rules, the road constraints provide strong cues and additional structure to the environment. These cues can be used to improve the Tahoe's estimate of its own location in the road network,

as well as the location and threat level of obstacles in its nearby environment. These additional cues also allow the Tahoe to plan in a simpler, more restricted world when all agents obey the rules of the road, ensuring more stable, robust response at the output of the path planner. It is the job of the scene estimator to identify and take advantage of these cues in order to enable the Tahoe to operate in this structured regime as often as possible.

The scene estimator acts as the interface between the sensors/Local Map and the intelligent planning hierarchy: it is the sensor fusion scheme that reduces the copious sensor data into the smallest pieces of information the path planner requires for normal driving. The scene estimator takes into account the constraints and assumptions of the Urban Challenge, whereas the Local Map and pose estimator make no simplifying assumptions. Along these lines, there are two observations that drive the development of the scene estimator. First, the DUC is, in most circumstances, a very constrained problem. Cars are the only moving objects on the road, and they will most often drive on the road. More importantly, all waypoints and checkpoints in the route network are guaranteed to be surveyed accurately with a military grade GPS, and they will be in easily-reachable locations. Stop line locations are also guaranteed to be surveyed accurately. The scene estimator is built to take advantage of these attributes. A second observation driving the development of the scene estimator is the requirement of the intelligent planning system to consider a reduced order environment model for planning, when possible. In other words, many obstacles found in the Local Map do not need to be considered in path planning. Examples include obstacles that are clearly not moving and not on the road. In fact, under normal driving circumstances the Tahoe only cares about obstacles and their lane positions on the one-dimensional road; it is the goal of the scene estimator to achieve this dimensional reduction in a statistically justified manner.

The Scene Estimation Algorithm

The scene estimator is divided into three parts: a map layer, a transient obstacle layer, and a persistent obstacle layer. Each layer is defined by the type of information each layer generates for the path planner. The map layer is built to take advantage of the position cues hidden in the road sensors, the route network file, and the unique attributes of the DUC. In essence, the map layer relies on Bayesian techniques to combine external position cues with the output of the pose estimator to generate a posterior estimate of the Tahoe's position within the RNDF that is more accurate than the pose estimator or the position cues alone. More specifically, the algorithm draws on two areas of active research: robotic simultaneous localization and mapping (SLAM) techniques, and GPS map aiding [16]-[20]. The goal in the map layer is to combine these techniques to simultaneously estimate the joint distribution:

$$p\left(X_{1:k}, W_{1:k}, L_{1:k}, M_{1:k} \middle| Z_{1:k}\right)$$

where $X_{1:k}$ is the Tahoe's X, Y position and heading within the route network, $W_{1:k}$ is which waypoint the Tahoe is tracking in the route network, $L_{1:k}$ is the Tahoe's lane estimate, $M_{1:k}$ is the route network itself. The available measurements $Z_{1:k}$ now include the RNDF itself, the output of the stop line detection algorithm, and the output of the MobilEye lane finding software, and the position and heading estimates of the pose estimator. For the purposes of the scene estimator, the combination $\left[X_{1:k}, W_{1:k}, L_{1:k}\right]$ is called "road network pose".

Notice that in selecting this vector of states to estimate, the problem has explicitly been condensed from free motion in a three dimensional environment to planar constrained motion, and that any altitude error introduced through GPS has little effect on the vehicle's position within the route network. Additionally, by defining position in terms of a physical location as well as a lane and waypoints, the system implicitly sets out to estimate the Tahoe's position in the route network without explicitly constraining vehicle motion to the road. In this way, the approach differs from GPS map aiding techniques which assume a human driver keeps the vehicle on the road at all times. Unfortunately, the desired posterior distribution presented above is too complicated to estimate in its full joint form, as vehicle dynamics are nonlinear and the route network is far from Gaussian. Instead, Team Cornell uses a Bayesian factorization similar to the Local Map and uses FastSLAM to separate the posterior into vehicle road network pose and the map [16]:

$$p\left(X_{1:k}, W_{1:k}, L_{1:k}, M_{1:k} \middle| Z_{1:k}\right) = p\left(X_{1:k}, W_{1:k}, L_{1:k} \middle| Z_{1:k}\right) \cdot p\left(M_{1:k} \middle| X_{1:k}, W_{1:k}, L_{1:k}, Z_{1:k}\right)$$

That is, the posterior has been separated into two problems: the problem of estimating vehicle road network pose $p\left(X_{1:k}, W_{1:k}, L_{1:k} \middle| Z_{1:k}\right)$, and the problem of estimating updates to the RNDF through $p\left(M_{1:k} \middle| X_{1:k}, W_{1:k}, L_{1:k}, Z_{1:k}\right)$ given the vehicles road network pose. The latter of the two problems is similar to the SLAM problem, except the waypoints in the RNDF $M_{1:k}$ are estimated instead of physical landmarks. This estimation problem allows Team Cornell's vehicle to augment the RNDF with additional waypoints if the RNDF waypoints are sparse; in other words, it allows the Tahoe to learn the map as it drives. In this case, the states $M_{1:k}$ represent road heading at each DARPA supplied waypoint, *and* road heading at each Tahoe supplied waypoint as it drives. As in FastSLAM, these road headings are estimated with independent Kalman Filters, one for each waypoint, conditioned on vehicle pose. Note, however, that the traditional landmark-based SLAM problem is specifically not used because Team Cornell feels there are enough opportunities for incorrect landmark identification that traditional SLAM will be too brittle for the DUC.

The primary estimation problem within the map layer of the scene estimator is the road network pose estimation problem, $p\left(X_{1:k}, W_{1:k}, L_{1:k} \middle| Z_{1:k}\right)$. Although the Tahoe's position within the network $X_{1:k}$ may be well-described by a linear Gaussian system with vehicle odometry, the multi-modal ambiguity of the lanes at an intersection $L_{1:k}$ suggests it is safer to estimate the entire joint distribution with a standard particle filter. Team Cornell implements this filter in two steps. First, road network pose particles are drawn randomly based on vehicle odometry and previous vehicle pose, then lane and waypoint selections are made randomly based on current vehicle pose and previous lane and waypoint selections. Particle weights are updated using available measurements: position and heading from the pose estimator (a comparatively weak global cue), stop line detections (a very strong but sporadic cue), and lane detections (a strong local cue). These measurements incorporate global pose information with lane cues to produce smaller uncertainties in vehicle position in relation to the map.

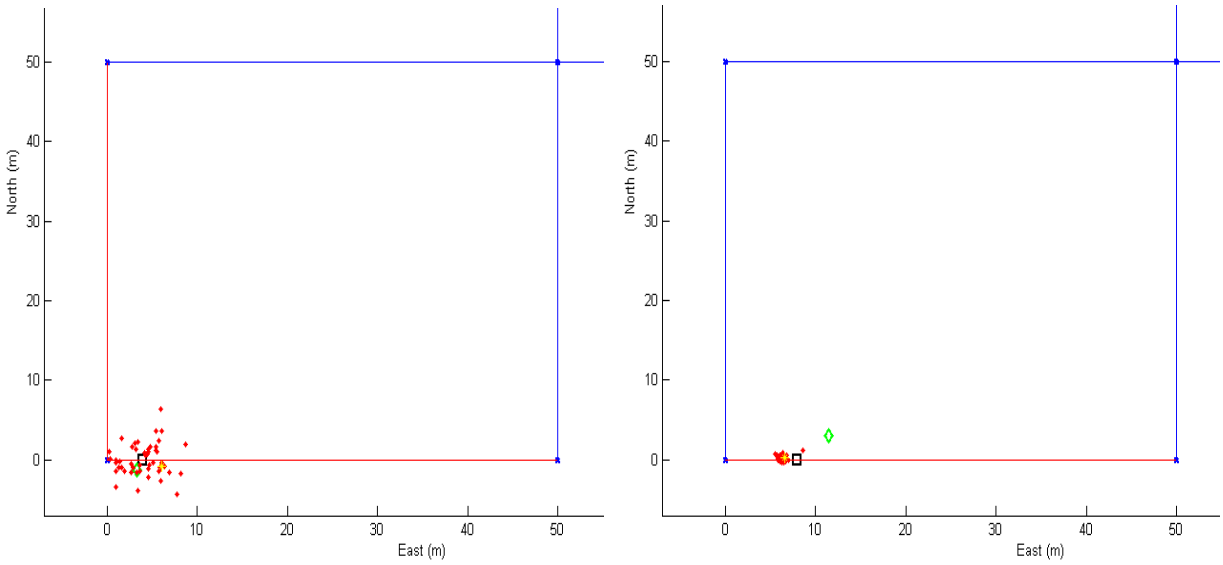

Figure 11: Left: vehicle pose estimate shortly after the start of a simulation. The vehicle pose particles (red) are widely dispersed near the true vehicle (black), and the selected road segments (red) are uncertain. The minimum mean-squared error estimate of the vehicle (yellow), has ~2m error. The GPS pose estimate is shown in green. Right: vehicle pose estimate after incorporating road information. With road measurements, all pose particles (red) are tightly clustered near the vehicle's true position, even though the GPS measurement (green) has significant error.

Figure 11 (left) shows the map layer of the scene estimator near the beginning of a sample simulation. Notice that the pose particles (red) are widely spread out over nearly 10m due to the limited capabilities of GPS. The particles do not even agree on what road the Tahoe is traveling, as several believe the vehicle is traveling north. The minimum mean square error estimate (yellow) is ~2m from the actual vehicle position. The estimate improves substantially when simulated road measurements are added in Figure 11 (right). The additional position cues provided by the road measurement and the route network reduce vehicle pose error from 2m (primarily due to GPS error) to <1m (primarily due to along-track ambiguity). This posterior error is further constrained each time the vehicle makes a turn, producing a pose estimate that is better than filtered GPS alone.

Once accurate road network pose and map estimates are generated, the scene estimator can determine the threat level of all the obstacles in the Local Map. This is done with a series of hypothesis tests, which combine the vehicle pose estimate with the Local Map relative obstacle states to determine which obstacles are on the road and what lane they are in. This step is accomplished by combining the Tahoe pose estimate with relative obstacle estimates from the Local Map into obstacle positions in the route network, and then comparing those positions to the route network map estimate for the location of each lane. By determining which obstacles are actually moving on the road, the scene estimator condenses the information from the Local Map into a simpler one-dimensional planning problem for the path planner. The scene estimator can also perform hypothesis tests to see whether all lanes are blocked, which might indicate the presence of a more permanent road block. A Hidden Markov Model (HMM) with states "road blocked" and "road not blocked" could be adapted to maintain the locations of permanent road blocks.

Intelligent Planning

The purpose of the intelligent planning system is to use the scene estimator's probabilistic estimates of the environment to determine vehicle actions. The system is designed from three basic requirements. First, the planner must be expandable such that additional behaviors can easily be added as the system matures. Second, the planner must accommodate asynchronous operation, so low level control loops can run much faster than high level trajectory planning. Finally, the system must be stable and robust despite potentially unpredictable actions of other vehicles in the environment.

Team Cornell's intelligent planner, summarized in Figure 12, addresses each of these requirements. Briefly, the algorithm consists of four steps: information retrieval, global objective planning, local trajectory planning, and local trajectory tracking. At the highest level, a monolithic belief base contains all environment data and state information transmitted by the scene estimator, as well as other parameters describing the current state of the Tahoe's planning algorithm. The belief base also contains semi-permanent and slowly changing information, such as information about the original RNDF and the locations of any discovered road blocks. This information is retrieved by a messaging service and delivered to the path planner at the beginning of each planning cycle. The information is condensed in the planner entry, where only threatening and pertinent obstacles are presented to the three layers of the path planner.

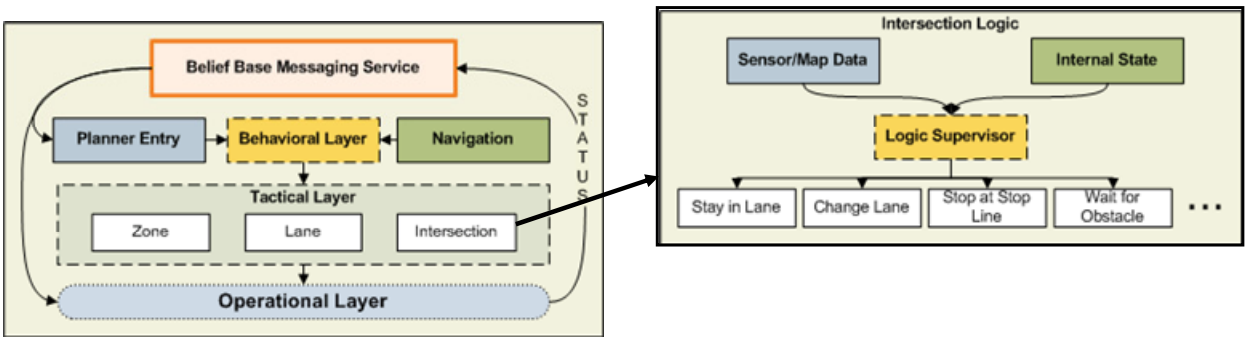

Figure 12: Left: The layers of Team Cornell's intelligent planning architecture (behavioral, tactical, and operational), and their connections to other components. Right: Expansion of the Intersection tactical component.

After the environmental information has been retrieved, the behavioral and tactical layers utilize it to generate a trajectory to follow. First, the behavioral layer uses the environment data to determine which of several tactical behaviors will be used in the tactical layer at the present iteration. Three representative behaviors, "intersection," "lane following," and "obstacle avoidance within a zone" are shown in Figure 12. Once selected, the appropriate tactical behavior is run with the condensed belief base data in order to generate a trajectory and speed. This trajectory is handed to the operational layer, which follows the trajectory until the next planning cycle has been completed and a trajectory is provided. This design allows the operational layer to be run much faster than the tactical layer to preserve vehicle stability.

Behavioral Layer

The behavioral layer functions as a global situation analyzer. First a traditional search technique is used to determine the minimum time global path to the next mission checkpoint, taking into account a priori RNDF information as well as semi-permanent obstacle locations and

even transient traffic jams. Figure 13 shows an experimental example of one such path, where a new path is planned when a road block is discovered. Once the desired global route is chosen, a situational analysis is performed to determine how to plan a trajectory over the portion of the global environment nearest to the Tahoe.

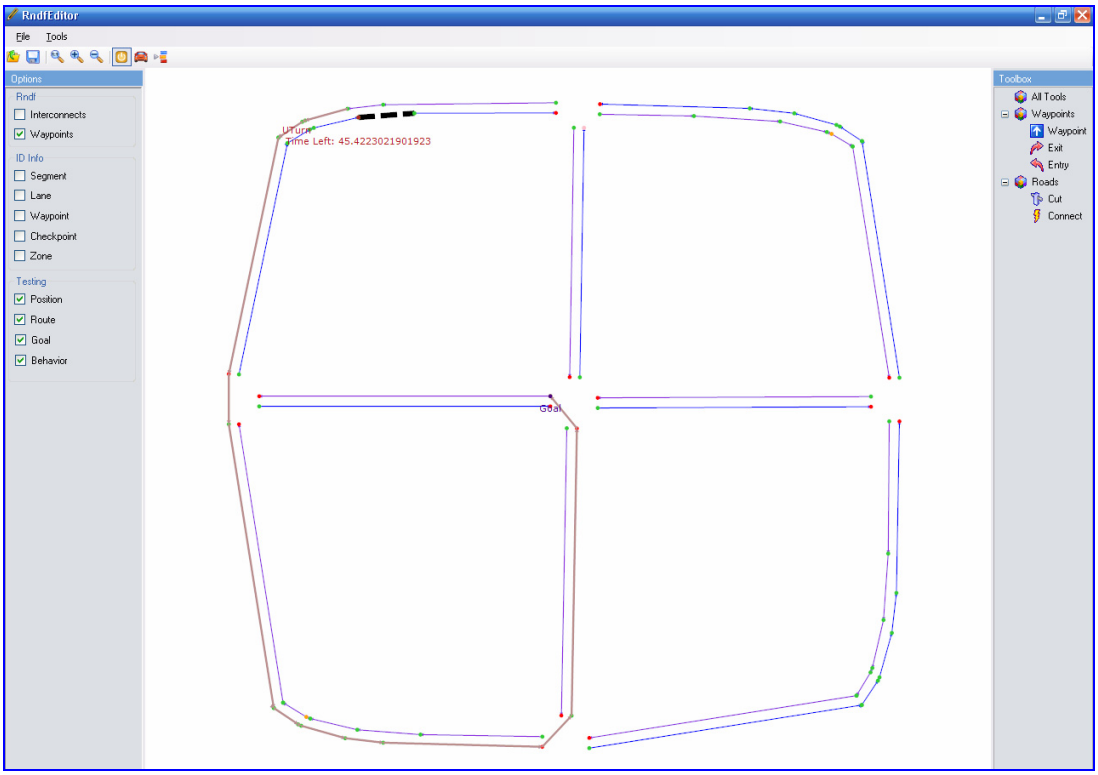

Figure 13: The behavior layer plans a new global route when a road block is discovered.

It is pointed out that although the behavioral layer nominally attempts to minimize time of travel to the next mission checkpoint, the optimality of the route is neither guaranteed nor desired. Because stability is far more desirable than optimality, previously discovered information must be kept within the belief base to prevent the system from continuously making the same bad decision. Instead, robustness to uncertainty in travel time, road blocks, and previous bad paths are handled using a probabilistic evaluation of each routing decision [21]. Emphasis in planning is placed on obtaining paths that are "good enough" rather than "optimal."

Tactical Layer

The tactical layer is broken up into an ever-expanding set of components, each designed to mimic a single behavior for a specific DUC scenario. Many of these behaviors can be considered human-like, such as passing a vehicle, moving around an obstacle, and stopping at an intersection. These components interpret obstacle data to generate a piece of a trajectory appropriate for the situation presented. Three of the most commonly-used tactical components, lane reasoning, intersection reasoning, and zone reasoning, are discussed briefly below.

Lane reasoning is applied when the vehicle travels between intersections and zones. This component is responsible for analyzing the intentions of surrounding vehicles, determining when to change lanes in preparation for a turn or merge, and avoiding temporary lane blockages. These

behaviors are accomplished by evaluating a small set of possible maneuver parameters (speed up, slow down, change lanes, etc.) for safety and feasibility. Each maneuver is initially checked with simple decision rules to determine whether it has a high probability of generating a collision or violating the rules of the road. If a maneuver fails this first test, a more in-depth Monte Carlo analysis of the scene is performed. Possible future behaviors of other vehicles (both moving and stopped) are sampled according to a Dynamic Bayesian Network (DBN), and random evolutions of the scene are checked against the proposed maneuver to determine the expected risk [22]. A "good enough" maneuver balancing risk and time-to-goal is then selected for implementation.

Intersection reasoning is invoked when the vehicle has stopped at an intersection. The first step is to use the output of the scene estimator to determine whether any objects are present at the intersection when the Tahoe arrives. If so, a DBN is used to classify each object in the intersection as a permanent blockage or a normal vehicle. These DBNs result in an evolving interpretation of the queue at the intersection, which the Tahoe uses to decide when to proceed [23]. This reasoning applies to four-way stops as well as T-intersections and other more complex scenarios simply by modifying the states of the DBNs used to evaluate lane occupancy. The DBNs also allow the Tahoe to cope with transient breakdowns in the rules of the road. For example, if the intersection becomes deadlocked for more than ten seconds, the Tahoe will initiate a randomized maneuver to edge carefully into the intersection. The DBNs classify the responses of other objects at the intersection, allowing the Tahoe to resolve situations safely.

Finally, zone reasoning applies when the Tahoe must navigate an unstructured environment to achieve a particular position and orientation. To accomplish this case in a somewhat organized manner, a basic lane structure is artificially imposed around the outside of the zone and leading to the desired parking spot. Within this semi-structured environment, a velocity obstacle path planner determines an appropriate velocity and heading command to avoid hitting any obstacles while attempting to follow the desired lane [24].

Operational Layer

Once the tactical layer has chosen a small trajectory to track, the operational layer converts that trajectory into steering, brake, throttle, and transmission commands. These commands are divided into two modules, a steering module and a speed module.

The steering module attempts to drive vehicle off-track error and heading error to zero based on a traditional linear quadratic regulator (LQR) controller. Most importantly, off-track and heading errors are calculated with respect to the desired trajectory, which is in turn represented with respect to the vehicle. In this way, the path itself is only loosely dependent on GPS, with heavier dependency placed on the more accurate differential motion produced by the pose estimator to transform the path from one time step to the next. Within the controller, path commands are fed back from off-track and heading errors in terms of a desired curvature (instantaneous trajectory), which is then transformed backwards into a desired steering wheel angle. Desired steering wheel angle is fed directly to the steering actuator.

Speed control, in contrast, is handled in two stages. First, a desired acceleration is computed. For stopping or deceleration maneuvers, this desired acceleration is computed as a constant deceleration over a specified distance. For vehicle following and speed tracking, a proportional-integral controller generates desired acceleration proportional to desired speed and desired following distance. Once the desired acceleration is computed, speed control is executed via feedback linearization to cancel vehicle and engine drag and determine desired brake and throttle accelerations. These accelerations are transformed into desired engine torque and master cylinder pressure, which are sent to the throttle and brake actuators, respectively.

Finally, collision avoidance is also implemented at the operational layer as a safety reflex. This reflex planner runs each time the operational layer determines actuator commands, making small perturbations to the commanded trajectory if safe clearance on any side of the Tahoe is not preserved. Although this collision avoidance considers only the nearest clusters of points detected by the Ibeos, it provides, at the very least, a basic greedy collision avoidance scheme able to function even if all the high level reasoning and sensor fusion should fail. Figure 14 shows experimental results of the operational layer guiding the vehicle along a trajectory (left) and avoiding clusters of points detected by the Ibeos (right).

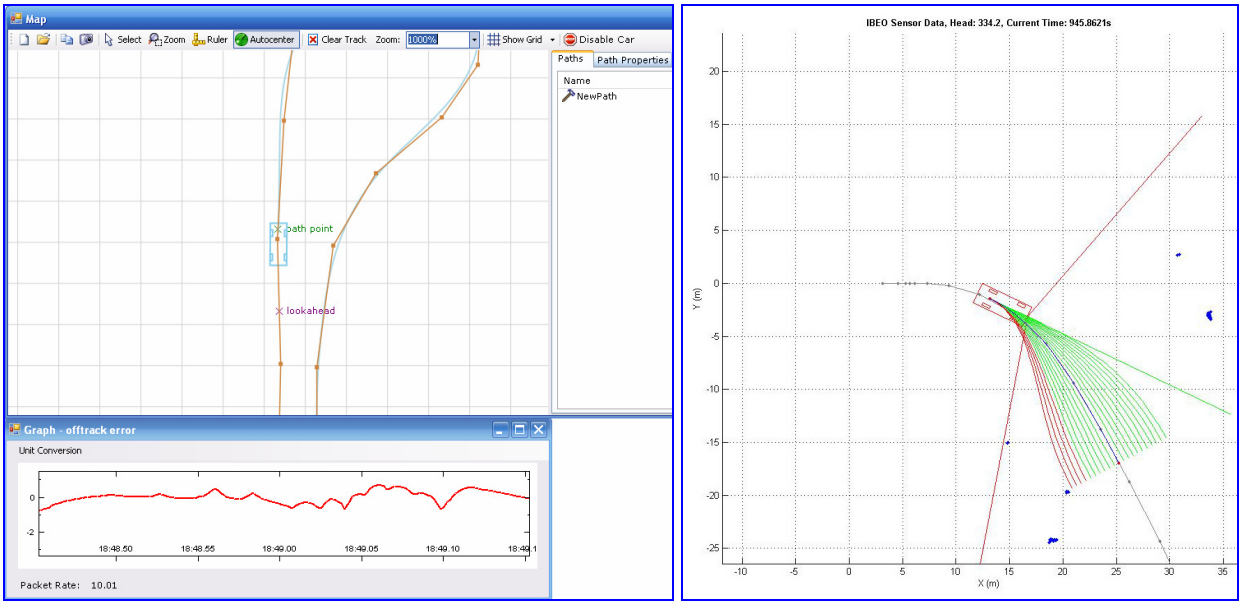

Figure 14: Operational layer implementation on the Tahoe. Left: GUI snapshot showing the monitoring of the vehicle during experimental testing; real time path/waypoint accuracy as well as vehicle health are evaluated. Right: internal implementation of the evasive planner against Ibeo data with path development and selection; the red paths are removed because of near collisions.

Experimental Implementation

The operational layer, in conjunction with the competition Tahoe and actuation hardware, has logged over 50 miles of autonomous driving at the time of this writing; Figure 14 shows a sample of these results. The behavior layer has been implemented in full real time simulation. Due to the expandability of the tactical layer, Team Cornell is and will be focusing on the tactical layer until the time of competition, using validation testing and software updates.

Conclusions

Team Cornell is currently developing and validating a full solution for autonomous urban driving for the 2007 DARPA Urban Challenge. This solution includes novel technologies in vehicle actuation, sensing architectures, local obstacle tracking, scene estimation, and intelligent planning. These technologies have been integrated into the competition vehicle; the current focus is on experimental validation in an urban setting. The systematic approach that Team Cornell has taken, including its Bayesian optimal approaches at multiple levels, lead to an important, highly probable solution to the complex problem proposed in the 2007 DARPA Urban Challenge

References

[1] Melihen, A., and Nastro, L. "Race for the Prize: Beating the Challenges of Robot Racing." Traffic Technology International, Dec., 2005 / Jan., 2006, pp. 70 – 72.

[2] Bierman, G. "Factorization Methods for Discrete Sequential Estimation." New York: Academic Press, 1977.

[3] Savage, P. "Strapdown Inertial Navigation Integration Algorithm Design Part 1: Attitude Algorithms." *J. of Guidance, Control, and Dynamics*, Vol. 21, No. 1, Jan.–Feb., 1998.

[4] Savage, P. "Strapdown Inertial Navigation Integration Algorithm Design Part 2: Velocity and Position Algorithms." J. of Guidance, Control, and Dynamics, Vol. 21, No. 2, Mar.–Apr., 1998, pp. 208 – 221.

[5] Lundberg, J., and Schutz, B. Recursion Formulas for Legendre Functions for Use with Nonsingular Geoptential Models. Journal of Guidance, Control, and Dynamics, Vol. 11, No. 1, Jan.–Feb., 1988, pp. 31 - 38.

[6] Bar-Shalom, Y., Rong-Li, X., Kirubarajan, T. "Estimation with Applications to Tracking and Navigation: Theory, Algorithms, Software." New York: John Wiley & Sons, 2001.

[7] Miller, I., Lupashin, S., Zych, N., Moran, P., Schimpf, B., Nathan, A., and Garcia, E. "Cornell University's 2005 DARPA Grand Challenge Entry." *Journal of Field Robotics*, Vol. 23, No. 8, 2006, pp. 625 – 652.

[8] Psiaki, M., and Mohiudden, S. "Modeling, Analysis, and Simulation of GPS Carrier Phase for Spacecraft Relative Navigation." AIAA Guidance, Navigation, and Control Conference and Exhibit, San Francisco, California, 15 – 18 Aug., 2005.

[9] Saastamoinen, J. "Atmospheric Correction for the Troposphere and Stratosphere in Radio Ranging of Satellites." Geophysical Monograph Series, ed. Henriksen, S., Mancini, A., and Chovitz, B., Vol. 15, 1972, pp. 247-251.

[10] Davis, J., Herring, T., Shapiro, I., Rogers, A., and Elgered, G. Geodesy By Radio Interferometry: Effects of Atmospheric Modeling Errors on Estimates of Baseline Length. Radio Science, Vol. 20, No. 6, Nov.–Dec., 1985, pp. 1593 - 1607.

[11] Sleewaegan, J.-M., De Wilde, W., and Hollreiser, M. "Galileo AltBOC Receiver." Proceedings of ENC GNSS, May, 2004.

[12] Huttenlocher, Felzenszwalb . "Efficient Graph-Based Image Segmentation," Intl. Journal of Computer Vision, vol. 59, no. 2, Jan., 2004, pp. 167-181.

[13] Bar-Shalom, Y., and Fortmann, T. "Tracking and Data Association." Orlando: Academic Press, 1988.

[14] Särkkä, S., Vehtari, A., and Lampinen, J. "Rao-Blackwellized Particle Filter for Multiple Target Tracking." Information Fusion, Vol. 8 No. 1, Jan., 2007, pp. 2 – 15.

[15] Miller, I., and Campbell, M. "Rao-Blackwellized Particle Filtering for Mapping Dynamic Environments." IEEE International Conference on Robotics and Automation, Rome, Italy, Apr., 2007.

[16] Montemerlo, M., and Thrun, S. "FastSLAM: A Factored Solution to the Simultaneous Localization and Mapping Problem." AAAI National Conference on Artificial Intelligence, Edmonton, Canada, 2002.

[17] Cui, Y., and Sam Ge, Shuzhi. "Autonomous Vehicle Positioning with GPS in Urban Canyon Environments." IEEE Trans on Robotics and Autom, Vol. 19, No. 1, Feb., 2003.

[18] Lee, K., Wijesoma, W., and Ibañez-Guzmán, J. "Map Aided SLAM in Neighbourhood Environments." IEEE Intelligent Vehicles Symp, Parma, Italy, Jun., 2004, pp. 836 – 841.

[19] Wijesoma, W., Lee, K., and Ibañez-Guzmán, J. "Motion Constrained Simultaneous Localization and Mapping in Neighbourhood Environments." International Conference on Robotics and Automation, Barcelona, Spain, Apr., 2005, pp. 1085 – 1090.

[20] El Najjar, M., and Bonnifait, P. "A Road-Matching Method for Precise Vehicle Location Using Belief Theory and Kalman Filtering." Autonomous Robots, Vol. 19, 2005, pp. 173 – 191.

[21] Ferguson, Stentz, and Thrun. "PAO* for Planning with Hidden State." Proceedings of the IEEE International Conference on Robotics and Automation, 2004.

[22] Broadhurst, Baker, and Kanade. "Monte Carlo Road Safety Reasoning." IEEE Intelligent Vehicle Symposium, Las Vegas, NV. pp. 319-324.

[23] Wang, Kearney, Cremer, and Willemsen. "Steering Behaviors for Autonomous Vehicles in Virtual Environments." Proc. of IEEE Virtual Reality, Mar., 2005, pp. 155 – 162.

[24] Fiorini and Shiller. "Motion Planning in Dynamic Environments Using Velocity Obstacles." Int'l Journal of Robotics Research, Vol. 17, No. 7, Jul., 1998, pp. 760-772.