Team Cornell's Skynet: Robust Perception and Planning in an Urban Environment

Isaac Miller and Mark Campbell
Sibley School of Mechanical and Aerospace Engineering
Dan Huttenlocher and Frank-Robert Kline
Computer Science Department
Aaron Nathan, Sergei Lupashin, and Jason Catlin
School of Electrical and Computer Engineering
Brian Schimpf
School of Operations Research and Information Engineering
Pete Moran, Noah Zych, Ephrahim Garcia, Mike Kurdziel, and Hikaru Fujishima
Sibley School of Mechanical and Aerospace Engineering
Cornell University
Ithaca, New York 14853
e-mail: itm2@cornell.edu, mc288@cornell.edu, dph@cs.cornell.edu, amn32@cornell.edu, fk36@cornell.edu, pfm24@cornell.edu, ncz2@cornell.edu, bws22@cornell.edu, sv15@cornell.edu, eg84@cornell.edu, jac267@cornell.edu, msk244@cornell.edu, hf86@cornell.edu

Received 30 January 2008; accepted 16 June 2008

Team Cornell's Skynet is an autonomous Chevrolet Tahoe built to compete in the 2007 DARPA Urban Challenge. Skynet consists of many unique subsystems, including actuation and power distribution designed in-house, a tightly coupled attitude and position estimator, a novel obstacle detection and tracking system, a system for augmenting position estimates with vision-based detection algorithms, a path planner based on physical vehicle constraints and a nonlinear optimization routine, and a state-based reasoning agent for obeying traffic laws. This paper describes these subsystems in detail before discussing the system's overall performance in the National Qualifying Event and the Urban Challenge. Logged data recorded at the National Qualifying Event and the Urban Challenge are presented and used to analyze the system's performance. © 2008 Wiley Periodicals, Inc.

1. INTRODUCTION

Team Cornell's Skynet, shown in Figure 1, is an autonomous Chevrolet Tahoe built to compete in the 2007 DARPA Urban Challenge. Skynet was built and developed at Cornell University by a team composed primarily of members returning with experience from the 2005 DARPA Grand Challenge. The team remained small, with 12 core members supported by 9 part-time contributors. They included professors, doctoral and master's candidates, undergraduates, and Cornell alumni. The team was selected from a grant proposal as one of 11 research-oriented teams to receive funding from DARPA to compete in the Urban Challenge. Additional support was gained through corporate sponsors, including Singapore Technologies Kinetics, Moog, Septentrio, Trimble, Ibeo, SICK, MobilEye, The Mathworks, Delphi, and Alpha Wire.

Team Cornell's development cycle proceeded as a carefully monitored research and engineering endeavor. The team ethos is one of maintaining knowledge across generations of researchers, and the system was largely developed and tested amid lessons learned from the team's participation in the 2005 DARPA Grand Challenge (Miller et al., 2006). The entire team designed and reviewed the system in weekly research meetings. As the system matured to testing, the entire team relocated to the Seneca Army Depot in Romulus, New York, where Team Cornell built and maintained a private road network for autonomous vehicle testing. Autonomous testing was conducted formally at the Seneca Army Depot, with a safety rider selected to monitor Skynet's decisions from the driver's seat for each test. Tests were coordinated between the safety driver, traffic drivers, and developers through the use of two-way radios. A remote emergency disabling switch was also actively maintained from a support vehicle during each test. In general, the system was brought online cautiously: development began in simulation, progressed to evaluation with sensor data logs, then to unit testing with actuators disabled, and finally to full closed-loop autonomous driving.

The remainder of the paper describes the system architecture, components, and performance in the Urban Challenge, all in the context of lessons learned from the Grand Challenge. Section 2 begins with a description of the Skynet's system architecture and timing and interface protocols. Section 3 continues with a detailed algorithmic description of each major subsystem in Skynet. Section 4 presents results for individual systems tested at the Urban Challenge National Qualifying Event (NQE). Section 5 presents general results and performance in the Urban Challenge Final Event, with several case studies used to highlight unique scenarios. Section 6 concludes with a review of lessons learned in the 2005 Grand Challenge, as well as new lessons and research questions posed by the results of the Urban Challenge.

2. SYSTEM ARCHITECTURE AND DATA FLOW

The general system architecture for Team Cornell's Skynet is shown in Figure 2 in the form of key system

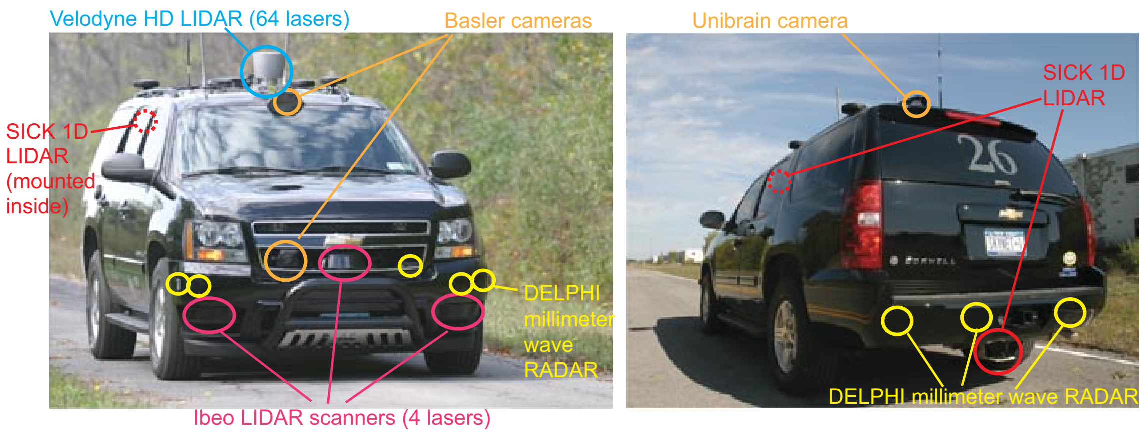

Velodyne HD LIDAR (64 lasers)
Basler cameras
SICK 1D LIDAR (mounted inside)
DELPHI millimeter wave RADAR
Ibeo LIDAR scanners (4 lasers)
Unibrain camera
SICK 1D LIDAR
DELPHI millimeter wave RADAR

Figure 1. Team Cornell's Skynet.

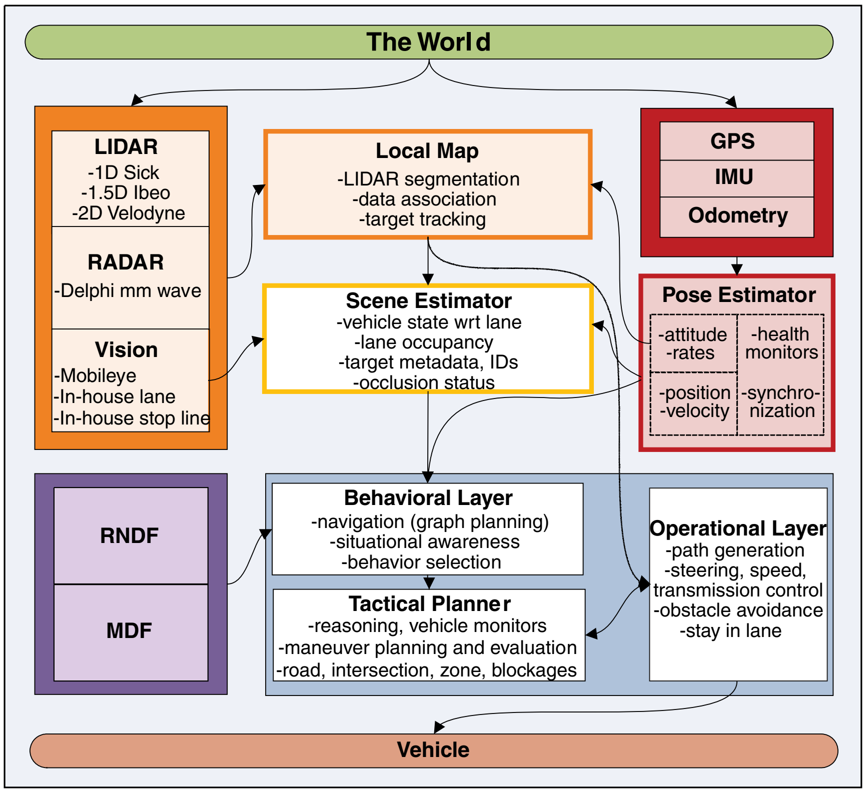


Figure 2. System architecture of Team Cornell's Skynet.

blocks and data flow. These blocks form the multilayer perception and planning/control solution chosen by Team Cornell to successfully drive in an urban environment. Details of each of these blocks are given in Section 3.

The world is perceived using two groups of sensors. Skynet itself is sensed using a combination of global positioning system (GPS) receivers, an inertial measurement unit (IMU), and vehicle odometry. These raw measurements are then fused in real time in the "pose estimator," producing tightly coupled position, velocity, and attitude estimates in an Earth-fixed coordinate frame. Skynet's environment, defined in the Urban Challenge as parked and moving cars, small and large static obstacles, and attributes of the road itself, is sensed using a combination of laser range finders, radar, and vision.

Two levels of probabilistic perception are used in Team Cornell's solution. The "local map" fuses laser, radar, and vision data, along with vehicle rotation rate and ground velocity measurements, to initialize, locate, and track static and dynamic obstacles over time. The local map tracks obstacles relative to Skynet, making no distinction between small/large or stationary/moving obstacles. The "scene estimator" then uses the raw object tracking estimates, pose estimates, and road cues from processed vision measurements in order to develop key statistics about Skynet and obstacles for the intelligent planner. Two sets of statistics are generated: those concerning Team Cornell's vehicle, such as its location with respect to the road and which lane it occupies, and those concerning other obstacles, including position/velocity, an identification number, lane occupancy, car likeness, and occlusion status based on other obstacles.

Planning and control for Skynet begins with road map and mission files: the route network definition file (RNDF) and the mission data file (MDF), respectively. These files are supplied by DARPA and required at the start of each mission. The RNDF lists all legally traversable lanes as ordered sequences of GPS way points; it also marks certain way points as "checkpoints," stop lines, and exit or entry points connecting multiple lanes. The MDF specifies an ordered series of checkpoints Skynet must achieve to complete its mission, as well as maximum speed limits Skynet may travel in each part of the RNDF.

Planning over the RNDF and MDF occurs at three layers of intelligent planning. The top-most "behavioral layer" uses the RNDF and MDF, along with obstacle information from the scene estimator and inertial estimates from the pose estimator, to interpret the environment and plan routes to achieve mission progress. The behavioral layer also decides which of four behavior states should be executed: road, intersection, zone, and blockage. Each of these four behaviors is then executed in the "tactical layer," where more finely detailed reasoning and planning occurs. The "operational layer," the lowest level of planning, produces a target path by smoothing an initial coarse grid-based path into one physically achievable by Skynet without violating constraints imposed by speed limits, road boundaries, and vehicle capabilities. The operational layer also has the responsibility of avoiding nearby obstacles, and it is therefore referenced only to Skynet itself. It uses no Earth-fixed information such as GPS positioning, similar to the local map. The interface between the tactical and operational layers can be iterative and is complex; this interface is described in more detail in Section 3.5.

The operational layer combines the target path defined by desired curvature and speed with models of Skynet to produce steering control (desired wheel angle), speed control (desired throttle and brake position), and transmission commands for Skynet. The commands are implemented by individual automation actuators (steering wheel, throttle, brake, transmission, and turn signals) on a stock sport utility vehicle (SUV) chassis. Standard aerospace motors are used for the wheel, brake, and transmission actuation, and the throttle is drive-by-wire through the stock General Motors CAN network.

Learning from system integration difficulties experienced in the 2005 Grand Challenge, Team Cornell addressed communication, synchronization, and data flow in its Urban Challenge entry before the constituent systems were designed. With the complex system architecture shown in Figure 2, three problems impact the design the most. First, there are simply massive amounts of data to be distributed between software and hardware modules around the car; approximately 76 Mb of data is transported across the car's networks each second. Second, the different software modules on the car each has various requirements on the level of synchronization and timing of the data necessary to satisfy their design requirements. Third, the many different types of sensors, actuators, and software modules each requires a custom interface design.

To overcome these challenges, Team Cornell developed and used a real-time data distribution network (RDDN) (Figure 3). The RDDN uses dedicated real-time microprocessors to interpret, time-stamp, and broadcast sensor data using the user datagram protocol (UDP) over a standardized Ethernet network. 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.

Specifically, the RDDN is composed of a 100-BaseT network of Motorola 9S12NE64 microcontrollers, each with embedded Ethernet MAC and PHY. Custom-built Ethernet-ready microcontrollers are interfaced to all sensors and actuators, including cameras, laser range finders, radars, the IMU, GPS receivers, the GM CAN network, and actuation controllers. The microcontrollers are synchronized with a single, master microcontroller with 0.1-ms accuracy using the IMU time stamp as a reference. Software modules are distributed on separate servers, each linked with switches. The modular microcontroller/server/switch system used a UDP multicast distribution of all sensor data, which enables universal availability of all data sources with accurate time stamping.

3. COMPONENT DESCRIPTIONS

This section provides algorithmic descriptions of the independent subsystems implemented in Team Cornell's Skynet. Discussion begins with a description of the vehicle actuation and power distribution system in Section 3.1. Section 3.2 continues with a description of Team Cornell's tightly coupled attitude and position estimator. Section 3.3 describes Skynet's obstacle detection and tracking system. Section 3.4 describes Skynet's algorithms for applying contextual information from the urban environment. Section 3.5 completes the discussion of Skynet's subsystems with a description of its intelligent planner.

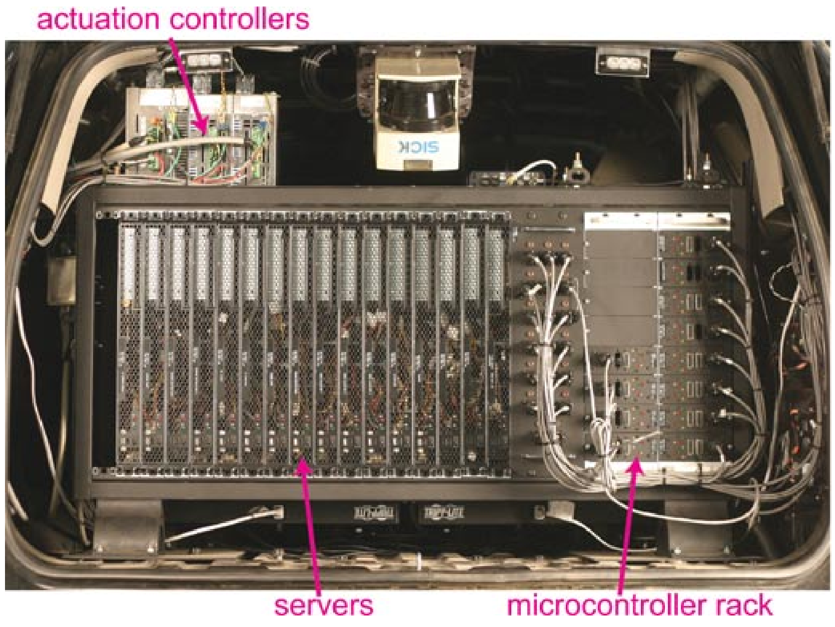


Figure 3. Real-time data distribution system, with servers and switches (left) and microcontrollers (right).

3.1. Vehicle Hardware

Team Cornell's Skynet is built on a 2007 Chevrolet Tahoe converted for autonomous operation. Selection of the Tahoe and its subsequent modifications was driven by two primary design requirements: responsiveness and reliability. Both design requirements are motivated directly by the environment of the Urban Challenge. An autonomous agent capable of fast response can adapt to potentially erratic behavior displayed by other robotic agents that might not be characteristic of a human-populated urban environment. The short development cycle of Urban Challenge vehicles also makes reliability a key requirement, because a vehicle that can be repaired quickly without specialized parts can get back to autonomous testing without much downtime. Team Cornell addresses these primary design requirements in four aspects of hardware selection: the vehicle chassis, the power subsystem, the drive-by-wire actuation, and the hardware packaging, each described below.

3.1.1. Chassis

Team Cornell's selection of the 2007 Chevrolet Tahoe as its vehicle chassis is based on design decisions intended to bolster responsiveness and reliability. One factor relevant to the decision is the fact that the Tahoe, as a full-size SUV, has sufficient room for a large number of computers, actuators, and human safety riders. Additionally, as an SUV it is larger and heavier than a typical sedan, making it more likely to survive low-speed collisions without serious damage. Also, the stock Tahoe has a large engine bay with provisions for auxiliary power generation, as Tahoes are prepared for conversion to emergency vehicles. Finally, the 2007 Tahoe comes equipped with a large set of easily accessible sensors, including stock throttle, odometry, and health-monitoring sensors, which are all used without modifying the Tahoe's electronics and throttle.

The commercially available Tahoe also addresses many reliability issues that affected Team Cornell's 2005 DARPA Grand Challenge entry (Miller et al., 2006). First, the manufacturer's warranty and stock parts ensure reliable operation, short lead time in acquiring replacement parts, and fast repair times. Second, the stock chassis has many rigid mounting points far inside the frame, allowing a computer rack to be mounted out of harm's way. Third, the 1,776-lb (806-kg) payload capacity ensures that the vehicle's performance is unaffected by large numbers of computers, batteries, and sensors. Finally, the common use of the Tahoe as an emergency vehicle makes a wide variety of commercially available after-market bumper, roll cage, and alternator kits available for additional reliability.

3.1.2. Power Subsystem

The design of Team Cornell's power subsystem was largely driven by lessons learned from the 2005 Grand Challenge. From that experience and a study of current computational hardware, initial power requirements were set at 2,400 W, with an additional 1,500 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 to inevitable minor power and equipment failures. Finally, the system was designed to switch readily between onboard power generation and wall power in the laboratory to permit extended algorithm testing without starting Skynet or using a generator.

Team Cornell used a separate power generator in the 2005 Grand Challenge, resulting in significant heat, noise, and reliability issues (Miller et al., 2006). In the Urban Challenge, the team opted instead to generate additional power with a secondary alternator, manufactured by Leece-Neville, mounted in the engine bay (Figure 4). The secondary alternator provides 200 A of peak output current at 24 V to drive a pair of Outback Power Systems inverters, mounted behind the front seats (Figure 4). Each inverter is capable of sourcing up to 3,500 W at peak. Both the alternator and the inverters exceed design requirements with a margin of safety. As a secondary system they also function independently from the stock electrical system on the Tahoe, so critical vehicle electronics do not compete with the computers and actuators for power. In addition, the design is redundant and can operate if one inverter fails. Finally, the inverters generate clean-enough power that sensitive electronics operate seamlessly, even during the transition from Skynet to wall power.

While the inverters provide the primary source of power to Skynet, they also charge four Optima deep-cycle automotive batteries mounted behind the inverters (Figure 4). These batteries are charged while the throttle is actively applied, and they are able to provide peak power even when the vehicle idles. The batteries also provide temporary power if the engine stops or Skynet needs to be restarted. The batteries act much like an uninterruptible power supply, but with an extended duration of approximately 6 h of reserve power.

3.1.3. Automation

The most significant decision affecting Team Cornell's development cycle was the choice to

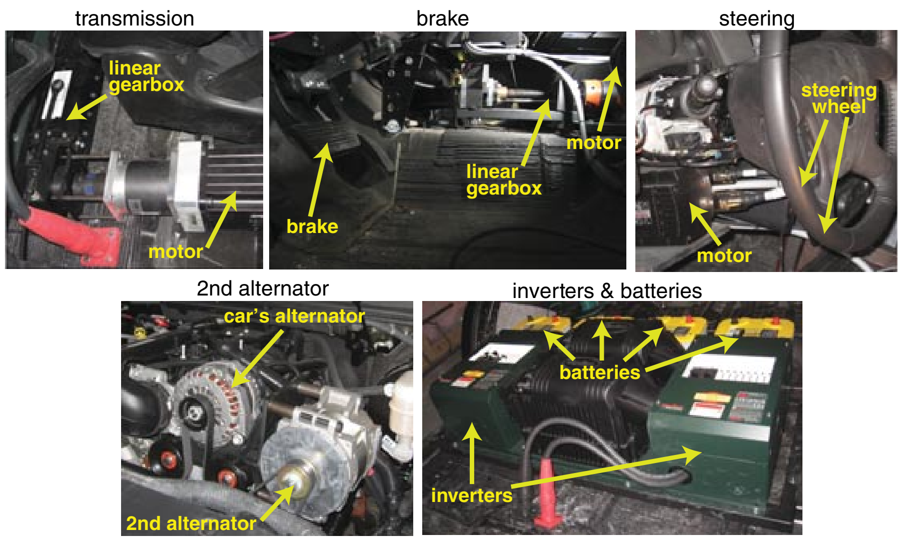


Figure 4. (Top) Team Cornell's custom actuation. (Bottom) Team Cornell's power generation and backup solution.

design and build the actuation scheme in-house for converting Skynet to drive-by-wire operation (Figure 4). This decision was made only after an extensive review of performance specifications, costs, and features of three commercially available alternatives: EMC, AB Dynamics, and Stahle. Three factors were considered critical in the decision. Scheduling was the first design constraint, and only the EMC conversion and in-house actuation were feasible within the Urban Challenge development cycle. The second factor was the ability to repair or modify the actuation quickly, which is necessary to resume testing rapidly in the event of a failure. This factor was perhaps most critical in the decision, as a failure in any commercial solution would require significant time spent transporting the vehicle to a factory for repairs. 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 or upgrading an in-house system far outweighed the time spent designing it.

Once Team Cornell decided to develop the vehicle actuation in-house, design specifications were created based on the most demanding maneuvers Skynet might need to perform during the Urban Challenge. In particular, a maximum steering angle performance requirement was defined as the ability to achieve a 700-deg change in steering wheel angle in 1 s. This requirement is taken from a standard National Highway Transportation and Safety Administration (NHTSA) fishhook maneuver, which is used to test commercial SUV rollover at 35 mph (56 km/h). Similarly, a braking force requirement was defined as the ability to achieve 100 lb of force of pedal pressure in 0.1 s, a maneuver defined as a Class A stop in the Consumer Braking Information Initiative. Vehicle modifications were also designed without compromising Skynet's human interface and safety systems. As a result of these design specifications, the actuation scheme places no additional restrictions on Skynet's maneuverability: limiting factors are equal to or better than those available to a human.

3.1.4. Packaging

Packaging hardware into Skynet was performed considering requirements of easy access, reliability, and impact survival. The front driver and passenger seats in Skynet remain unmodified to hold two passengers, one as a safety driver and one as a developer. The actuation scheme is also packaged into the front seats: the steering actuator is mounted in parallel with the steering column without affecting driver legroom, the brake actuator is mounted in the passenger leg area with a pull cable attached to the brake pedal, the throttle actuator is entirely electronic and built into a computer console between the front seats, and the transmission actuator is mounted behind the driver's seat with a push/pull cable attached to the transmission. The middle seats of Skynet are removed entirely and are replaced with the inverters and batteries for power generation and storage. A protective steel case covers both the batteries and inverters and allows easy access to the trunk. The third row of seats is also entirely removed and is replaced by Team Cornell's computer rack. The computer rack assembly mounts rigidly to Skynet's frame but is isolated from shock and vibration by four flexible mounting brackets provided by Barry Controls, Inc.

The rack itself is a steel frame custom built in-house to hold 17 single-rack-unit (1U) form factor computers, identical for rapid replacement. The rack is deliberately designed to house as many computers as possible, as its design was based on pessimistic overestimates of Skynet's computing requirements. In fact, Skynet fully utilizes only 7 of its 17 computers: 1 for position estimation, 1 for obstacle tracking, 1 for scene estimation, 1 for lane finding, 2 for constrained path planning, and 1 for high-level, rule-based path planning. The remainder of the computers run comparatively lightweight sensor processing algorithms. No attempt has been made to package these algorithms more efficiently, as Skynet has been designed as an expandable research platform. Each of Skynet's computers houses a dual-core Pentium Mobile laptop processor, with clock speeds ranging from 1.66 to 2.13 GHz and 2 Gb RAM. Laptop processors are used for heat and power savings, permitting the computers to be run safely in midday heat using Skynet's stock air-conditioning system. All computers run Windows Server 2003, selected due to the team's extensive experience with programming in the Windows environment. Potential Windows timing and thread scheduling problems have been avoided through the use of accurate microcontroller time stamping. Skynet's time-stamped Ethernet RDDN, discussed in Section 2.1, allows each process to maintain correct temporal ordering of data while tolerating common deviations in thread scheduling and execution times.

Cable control around Skynet was also considered in packaging the hardware during Skynet's autonomous conversion. All roof cables were routed together through a single waterproof (IP67) roof breach. Power cables are run on the floor of the car upward to power computers and microcontrollers in the rack, and Ethernet data cables flow down from the interior ceiling.

To improve survivability in the case of a collision, most sensors were mounted inside Skynet. Forward and rear-facing Ibeo and SICK laser range finders were all embedded into the front and back bumpers. All forward, side, and rear-facing radars were also placed behind the bumper plastic, which did not affect their performance. Forward and rear-looking cameras were mounted on the roof and behind Skynet's front grille. Side-mounted SICK laser range finders were also mounted inside Skynet, and the back door windows were replaced with infrared transparent plastic to allow detection and scanning from a safe vantage point.

3.2. Position, Velocity, and Attitude Estimation

Team Cornell's pose estimator fuses external satellite navigation signals with onboard sensing to supply the real-time position, velocity, and attitude (pose) solution used for absolute positioning. Team Cornell's pose estimator was designed and built with several objectives in mind. First, as the sole source of external absolute position information, the pose estimator must by itself meet the absolute positioning requirements of the Urban Challenge: driving within potentially unmarked lanes, stopping within 1 m of designated stop lines, and parking in potentially unmarked parking spaces. Each of these requirements mandates at least submeter positioning accuracy in the solution produced by the pose estimator. Second, the pose estimator must supply estimates of Skynet's differential motion to permit transformation of a vehicle-fixed coordinate frame over time. This requirement stems largely from deliberate design decisions made to reference all obstacles and path constraints in a vehicle-fixed coordinate frame, which is not affected by discontinuous changes in the absolute position solution as the quality of the GPS signal evolves over time. Finally, and most important, the pose estimator must produce a smooth and robust solution in the changing urban environment. It must be designed to cope with rapidly changing GPS satellite visibility, multipath and other signal distortion, and potential short-term GPS blackouts due to the presence of trees and buildings. In essence, biased, overconfident, and brittle pose estimates are all particularly fatal as the pose solution is ultimately used to determine which rules of the road apply at each vehicle planning cycle.

Although off-the-shelf pose estimators are sufficient to satisfy the design requirements, a cost analysis drove Team Cornell to develop a custom pose estimator in-house. In particular, Team Cornell already possessed all the necessary hardware and expertise from the 2005 DARPA Grand Challenge to implement the pose estimator in-house, so the tradeoff fell between time spent developing the custom solution and the up-front price paid to purchase a top-quality, off-the-shelf equivalent. In the end, better understanding of GPS signal behavior for debugging, the freedom to design a custom interface, and the ability to make filtering considerations specific to the Urban Challenge tipped the scales against off-the-shelf equivalents.

The pose estimator that evolved from these design considerations fuses information from four sensors: a Litton LN-200 IMU, Skynet's ABS wheel encoders, a Septentrio PolaRx2e@ GPS receiver, and a Trimble Ag252 GPS receiver. The LN-200 IMU is a combined three-axis fiber-optic rate gyroscope and a three-axis silicon accelerometer, rigidly mounted on the floor of Skynet, along its centerline, just above the rear axle. The LN-200 integrates its rate gyros and accelerometers to report vector measurements of changes in orientation and velocity of its internal IMU-fixed coordinate frame as a digital signal at 400 Hz. These measurements may then be integrated to determine the position and orientation of the IMU relative to an initial configuration. Skynet's stock ABS sensors, optical encoders mounted at each wheel, are also used to aid in this dead-reckoning integration scheme. Information from these encoders is retrieved over Skynet's stock GM CAN network at 30 Hz and is used to provide a measurement of vehicle speed to help slow IMU integration errors' rate of growth. The two GPS receivers, the Septentrio and the Trimble, are both used to keep the integration errors in the dead-reckoning scheme bounded when GPS signals are available. The Septentrio, a three-antenna, single-clock, 48-channel GPS receiver, provides raw pseudorange, Doppler shift, and carrier phase measurements of all visible GPS satellites on all its antennas at 5-Hz synchronized measurement intervals. Its antennas are mounted in an "L" pattern on Skynet's roof, as far apart as possible to promote greater observability in the differential signal between the antennas. The Septentrio also decodes the WAAS signal at the same rate to provide higher fidelity GPS error models. Finally, the Trimble, an agricultural-grade, single-antenna GPS receiver, is used solely to decode high-precision (HP) OmniSTAR differential corrections. These corrections are supplied at 10 Hz with an advertised accuracy of 10cm (Trimble, 2004). This HP signal, when statistically validated, is the primary source of submeter positioning information for satisfying the first-pose-estimator design requirement.

Team Cornell's pose estimator blends its four sensors in a tightly coupled estimator that utilizes the strengths of each sensor while compensating for their weaknesses with sensor diversity. The LN-200 provides the fast and accurate update rates necessary to produce a smooth and statistically robust solution, but it must be integrated and therefore suffers from integration errors. The wheel encoders slow the rate of growth of IMU integration errors but are subject to errors due to wheel slip. The Septentrio, although providing data at a much slower rate, corrects integration errors with absolute positioning information obtained from GPS and WAAS. These three sensors cannot consistently achieve submeter accuracy even when fused, but they can do so when supplemented with occasional updates from the HP signal received by the Trimble. The HP signal tends to be brittle, occasionally biased, and difficult to track for long periods of time, but statistical hypothesis tests using information fused over time from the other sensors is sufficient to determine when the HP signal is usable and when it is not. The resulting pose estimator, described below, uses these sensor strengths and diversity to satisfy the system's design requirements.

Team Cornell's pose estimator builds upon techniques and lessons learned from the pose estimator built for Cornell's 2005 Grand Challenge entry (Miller et al., 2006). Like the 2005 pose estimator, it collects and fuses information from its constituent sensors via an extended square root information filter (SRIF). The SRIF is a numerically robust implementation equivalent to the traditional Kalman filter (KF), but it achieves twice the numerical stability and precision by maintaining and propagating a square root of the state's information matrix instead of a state covariance matrix (Bierman, 1977). The SRIF also has the advantage of being able to correctly initialize state estimates with infinite covariance, which are simply represented in the SRIF as estimates with zero information. Beyond these two convenient features, the extended SRIF is functionally identical to a traditional extended Kalman filter (EKF). The SRIF is divided into familiar prediction and update steps, which, for this application, correspond to a dead-reckoning numerical integration step and a GPS, HP, or encoder measurement correction step.

Similarities with Cornell's 2005 pose estimator end with the SRIF. Numerical integration for dead reckoning and prediction is performed as a series of discrete Euler steps, each corresponding to a single change in orientation and velocity reported by the LN-200. Corrections for the Earth's rotation rate and coriolis and centripetal accelerations measured by the LN-200 are also made using Euler approximations in a method similar to that of Savage (1998a, 1998b). Velocity changes due to gravity are also subtracted from the integration using an Euler approximation, with the gravity vector calculated using a numerically stable Legendre polynomial construction algorithm and the full 360×360 EGM-96 gravity potential model (Lemoine et al., 1998; Lundberg & Schutz, 1988). Time-correlated GPS satellite range bias estimates, Septentrio receiver clock offset and drift rates, constant double-differenced carrier phase ambiguities, and constant-rate gyro and accelerometer biases are also filtered as part of the pose estimator, using the autocorrelated and random walk dynamic process models described by Bar-Shalom, Rong Li, and Kirubarajan (2001).

Asynchronous updates are performed within the pose SRIF as measurements become available from the wheel encoders, Septentrio, and Trimble. For wheel encoder measurements, a filter update is calculated from the measured speed of Skynet. For Trimble measurements, a filter update is calculated from the location of the Trimble, measured using the HP signal. For Septentrio measurements, the measured pseudoranges, Doppler shifts, and double-differenced carrier phases are all used to update the pose estimate in the SRIF on a satellite-by-satellite, antenna-by-antenna basis. This updating technique "tightly couples" the Septentrio to the IMU in the pose estimator by estimating vehicle pose, IMU biases, receiver clock errors, and satellite errors, all within a single centralized SRIF. It has the advantage that information is processed from each GPS satellite individually, allowing the filter to gain information even when fewer than four satellites are visible (Artes & Nastro, 2005). More important, it allows satellite signals to be modeled individually, so that each satellite can be statistically tested for significant errors and weighted in the SRIF according to its signal quality. Team Cornell's pose estimator takes advantage of this opportunity by expanding the broadcast GPS satellite signal model to include weather-based tropospheric corrections, time-correlated multipath models, and receiver thermal noise (Bar-Shalom et al., 2001; Davis, Herring, Shapiro, Rogers, & Elgered, 1985; Psiaki & Mohiudden, 2007; Saastamoinen, 1972; Sleewaegen, De Wilde, & Hollreiser, 2004). This expanded signal error model is then combined with previously fused pose estimates to perform a χ^2 hypothesis test on the entire set of GPS measurements to evaluate measurement validity (Bar-Shalom et al., 2001). If the set of measurements fails the hypothesis test, individual satellites are tested in an attempt to find a set of satellites with lower signal distortion. If no such set is found, the entire measurement is abandoned. Similar filter integrity monitoring hypothesis tests are performed on each wheel encoder measurement and each HP measurement to prevent measurements with significant errors from corrupting the filtered pose solution and disturbing vehicle behavior.

Team Cornell's pose estimator is implemented in C++ and runs dead-reckoning integrations at 400 Hz on a 64-bit dual-core Pentium-Mobile processor running Windows Server 2003. Full predictions of the square root information matrix are performed at 200 Hz due to the computational expense of a large QR-factorization mandated by the SRIF. The full-pose solution is reported at 100 Hz to all other systems in Skynet.

3.3. Obstacle Detection and Tracking

Team Cornell's obstacle detection and tracking system, called the local map, fuses the output of all obstacle detection sensor measurements over time into one vehicle-centric map of the local environment surrounding Skynet. The local map is built to satisfy four design requirements. First, diversity in both output information and placement of obstacle detection sensors around the car at a minimum requires a centralized algorithm to collect and fuse information to distinguish between traversable and occupied regions of the car's surroundings. Second, the local map needs to resolve conflicts between sensors. In this respect, mapping the sensors to a single coordinate frame is not enough; the sensors must be actively fused to a single interpretation of the environment for the planner to act upon. Third, the system must predict and track the motion of dynamic obstacles, to allow the planner to respond differently to moving traffic vehicles and static obstructions. Finally, the local map must be stable and robust over time, despite the rapidly changing urban environment. In particular, the local map must remain stable as other intelligent maneuvering agents pass into and out of view of various sensors in Skynet. Such stability is required to perceive complex intersection, merging, and traffic scenarios correctly as well, making it a critical requirement for success in the Urban Challenge.

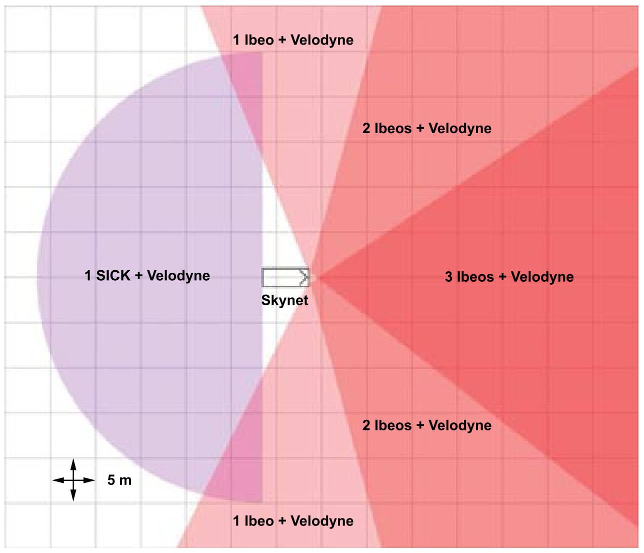
The local map fuses information from three sensing modalities: laser range finders, radars, and optical cameras. Team Cornell's Skynet is equipped with seven laser range finders: three Ibeo ALASCA XT range finders, two SICK LMS-291 range finders, one SICK LMS-220 range finder, and one Velodyne HDL-64E range finder. The three Ibeos, mounted in Skynet's front bumper, each returns range and bearing pairs taken from four separate laser beams at 12.5 Hz over a 150-deg field of view with angular resolution of approximately 1 deg. The two SICK 291s, mounted with vertical scan planes inside Skynet's rear doors, each returns range and bearing pairs over a 90-deg field of view with angular resolution of approximately 0.5 deg at 75 Hz. The SICK 220, mounted in Skynet's rear bumper with a single horizontal scan plane, returns range and bearing pairs over a 180-deg field of view with angular resolution of approximately 1 deg at 37.5 Hz. The Velodyne, mounted on the centerline of Skynet's roof above the front seats, returns range and bearing pairs from 64 separate laser beams over a 360-deg field of view at 15 Hz. Skynet is also equipped with eight Delphi FLR radars, each of which returns data for up to 20 tracked objects at 10 Hz. The Delphis are mounted five in the front bumper for forward and side-facing detection and three in the rear for backward detection. Finally, Skynet is equipped with one backward-facing Unibrain Fire-i 520b optical camera, mounted just above the trunk. This camera runs MobilEye SeeQ software that reports tracked obstacles at approximately 15 Hz. Table 1 summarizes Skynet's obstacle detection sensors, and Figure 5 gives a top-down view of Skynet's laser range finder and radar coverage.

Table I. Skynet's obstacle detection sensors.

Sensor	Location	Type	Rate, Hz	Field of view, deg	Resolution
Ibeo ALASCA XT	Front bumper left	Laser	12.5	150	1 deg
	Front bumper center	Laser	12.5	150	1 deg
	Front bumper right	Laser	12.5	150	1 deg
SICK LMS-291	Left back door	Laser	75	90	0.5 deg
	Right back door	Laser	75	90	0.5 deg
SICK LMS-220	Back bumper center	Laser	37.5	180	1 deg
Velodyne HDL-64E	Roof center	Laser	15	360	0.7 deg
Delphi FLR	Front bumper left (2×)	Radar	10	15	20 tracks
	Front bumper center	Radar	10	15	20 tracks
	Front bumper right (2×)	Radar	10	15	20 tracks
	Back bumper left	Radar	10	15	20 tracks
	Back bumper center	Radar	10	15	20 tracks
	Back bumper right	Radar	10	15	20 tracks
Unibrain Fire-i 520b	Back roof center	Optical	15	20–30	N/A

Like the pose estimator, the local map relies on sensor diversity to circumvent the shortcomings of each individual sensor. At the highest level, the laser range finders are most effective at determining obstacle position, but they do not measure obstacle speed. When combined with Delphi radars, which measure accurate speed (range rate) but poor range and bearing, the overall set of sensors yields accurate position and speed measurements of all obstacles near Skynet. This set of measurements is used within the local map to generate a central fused estimate of Skynet's surroundings, distinguishing traversable areas from those laden with obstacles, and identifying other moving agents.

To keep sensor interfaces with the local map simple, all sensor measurements are fused at the object level. That is, each sensor measurement is treated as a measurement of a single object, whether another



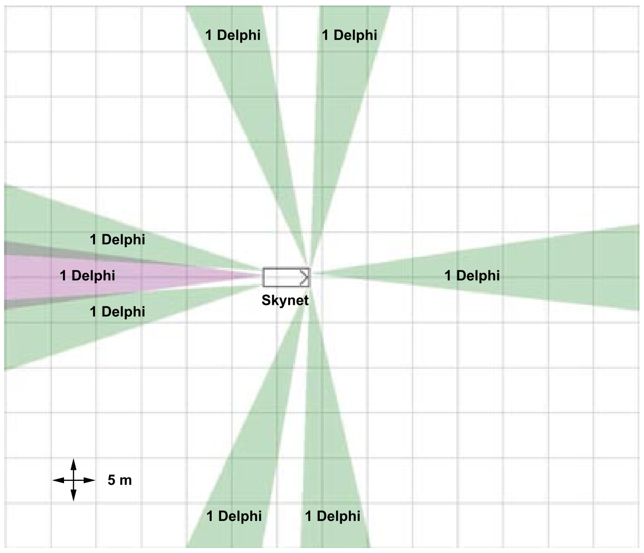


Figure 5. (Left) Laser range finder azimuthal coverage diagram for Team Cornell's Skynet. (Right) Radar azimuthal coverage diagram. Skynet faces right in both coverage diagrams. A rear-facing optical camera is not shown, nor are two laser range finders with vertical scan planes that detect obstacles immediately to the left and right of Skynet.

intelligent agent or a typical static obstacle. Both the Delphi radars and the MobilEye SeeQ software fitted easily into this framework, as their proprietary algorithms automatically transmit lists of tracked obstacles. The laser range finders, which all return lists of range and bearing pairs, are postprocessed to fit into this object-level framework. First, laser returns corresponding to the ground and any other objects too low to the ground to be vehicles are removed from the set of range finder points under consideration. This is accomplished through the use of a ground model constructed by rasterizing returns from each Velodyne frame into a time-filtered gridded ground map with techniques similar to those used in the 2005 Grand Challenge (Miller & Campbell, 2006; Miller et al., 2006). The primary difference between this method of ground modeling and the 2005 approach is that it represents the ground in a vehicle-fixed coordinate frame, using differential vehicle motion estimates from the pose estimator to keep the terrain grid fixed to Skynet as it moves.

Once ground and low obstacle laser returns are removed from consideration, a clustering step is performed to group laser returns into distinct objects for measurement. The clustering algorithm uses Euclidean distance thresholds to assign cluster membership, and it is run separately with thresholds of 0.5 and 1 m to generate clusters. Only clusters that are identical across both thresholds are considered distinct enough to be treated as measurable obstacles. The number of clusters is also further reduced by analyzing occlusion boundaries in the ranges of nearby points; only clusters that are completely visible are considered distinct enough to generate measurements. From there, each distinct obstacle cluster is used to generate measurements, which are then passed to the local map to be fused at the object level.

The local map fuses object measurements from its constituent sensors by treating the obstacle detection and tracking problem as the joint estimation problem of simultaneously tracking multiple obstacles and determining which sensor measurements correspond to those obstacles (Miller & Campbell, 2007). In the language of classical Bayesian estimation, the problem is formulated as estimating the joint posterior:

$$p[N(1:k), X(1:k) | Z(1:k)], \qquad (1)$$

where $N(1:k)$ is a set of discrete random variables indicating which sensor measurements correspond to which tracked obstacles at time indices 1–k, $X(1:k)$ is a set of continuous random variables representing the states of the obstacles being tracked at time indices 1–k, and $Z(1:k)$ are the full set of measurements from the first measurement frame to the current time index k. Note that the number of obstacles is also represented implicitly as a random variable in the cardinality of $N(k)$ and $X(k)$ at any particular time index and must also be determined in the local map's framework. In the Urban Challenge environment the joint estimation problem posed in Eq. (1) is generally too large to be solved by a direct brute-force estimator, such as a particle filter, evaluating hypotheses over the full multivariate state space. Instead, Team Cornell factorizes the posterior to yield two manageable components:

$$p[N(1:k)|Z(1:k)] \cdot p[X(1:k)|N(1:k), Z(1:k)], \qquad (2)$$

where, intuitively, $p[N(1:k)|Z(1:k)]$ describes the task of determining the number of obstacles to track and assigning measurements to those obstacles and $p[X(1:k)|N(1:k), Z(1:k)]$ describes the task of tracking a known set of obstacles with known measurement correspondences. In the local map, these two densities are estimated separately using a particle filter to draw hypotheses about measurement correspondences and banks of EKFs to track obstacles using each particle's hypothesis about measurement assignments (Miller & Campbell, 2007). This approach solves the joint estimation problem without wasting computational resources assigning particles over continuous obstacle states, which can be estimated effectively and inexpensively with standard EKFs.

Each EKF within a local map particle tracks one potentially moving obstacle under the measurement correspondence hypothesis of that particle. Each obstacle is modeled as a set of cluster points storing the geometric information known about that obstacle, and all those cluster points are referenced to an obstacle-fixed coordinate frame. The EKF then estimates the following states for each obstacle at time index k: the (x, y) location of the origin of that obstacle's coordinate frame, the angle ϕ orienting the obstacle's coordinate axes with respect to Skynet, the angle θ denoting the obstacle's heading relative to Skynet, and the obstacle's ground speed s. Obstacle maneuvers, i.e., changes in speed and heading, are modeled as random walks, with noise parameters set to encompass feasible accelerations and turning rates at Urban Challenge speeds. In the prediction step of an obstacle's EKF, these five state variables are numerically integrated with a fourth-order Runge–Kutta algorithm. This EKF prediction uses Skynet's differential motion estimates from the pose estimator to keep all tracked obstacles in a Skynet-centric coordinate frame. Although such a moving coordinate frame couples obstacle tracking errors to errors in Skynet's differential motion estimates, the quality of Skynet's IMU prevents this coupling from having any substantial impact on obstacle tracking. More important, the Skynet-centric coordinate frame ensures that absolute positioning errors, which depend heavily on the quality of the GPS environment, do not affect obstacle tracking at all.

The update step of each obstacle's EKF changes depending on the type of sensor providing the information. Both MobilEye and radar updates are performed as traditional EKF update steps with measurements consisting of a range, bearing, and range rate toward or away from the appropriate sensor. Updates from the side SICK 291s are scalar measurements consisting of distance from the side of Skynet to the obstacle. For the Ibeos and the rear SICK 220, three measurements are extracted from each clustered set of range finder points: bearings of the most clockwise and most counterclockwise points in the cluster and range to the closest point in the cluster. This set of measurements is chosen specifically for its favorable and stable behavior under linearization, though the linearization is never computed explicitly (Miller & Campbell, 2007). Instead, the update step utilizes the unscented transform from the sigma point filter (SPF) to compute an approximate linearization of the measurement about the current obstacle state estimate and the current cluster of points representing that obstacle (Julier & Uhlmann, 1997). In all update types, measurement noise is assumed to be additive and Gaussian. This allows measurement covariance matrices to be set according to the characteristics of each sensor or obstacle detection algorithm, while still ensuring proper updates to obstacle state covariance matrices through the use of EKF and SPF linearizations.

Each particle of the local map represents one complete hypothesis about the set of obstacles in Skynet's environment and which sensor measurements correspond to those obstacles. At each sensor frame, the local map's particles choose nondeterministically whether each sensor measurement corresponds to a new obstacle, should be assigned to the EKF of a specific existing obstacle, or should be ignored as clutter. These choices are made according to the likelihood that the measurement corresponds to a model of where new obstacles are likely to be seen, to existing obstacles, or to the class of false measurements commonly made by each sensor. The local map is resampled at the end of any sensor frame in which the effective number of particles is less than half the true number of particles (Arulampalam, Maskell, Gordon, & Clapp, 2002). The most likely particle and all its tracked obstacles are then broadcast at 10 Hz on Skynet's data network.

Like the pose estimator, the local map ensures that sensor measurements are processed in correct temporal order by keeping a queue of recent sensor measurements, sorted by age. The oldest measurements in the queue are used to update the local map once every sensor has either contributed a newer piece of data or timed out. This queuing structure, along with the rest of the local map, occupies a single core of a dual-core Pentium-Mobile processor running Windows Server 2003. It reports the current most likely list of obstacles at 10 Hz to Skynet's data network. It also maintains a set of laser range finder points confirmed to be obstacles but not tracked, either because they were deemed unstable during clustering or because they were too small to be moving vehicles. This list is also reported on Skynet's data network to ensure that the planner avoids all moving and static obstacles.

3.4. Environment Structure Estimation

Team Cornell's local map, described in Section 3.3, broadcasts a list of tracked and untracked obstacles at 10 Hz to Skynet's internal data network for collision avoidance. These obstacles are maintained in a vehicle-centric coordinate frame and are never referenced to absolute coordinates in the local map tracking scheme. This approach keeps absolute positioning errors from affecting obstacle avoidance, which depends only on the relative positioning of obstacles with respect to Skynet. This approach also avoids the potentially incorrect assumption that other moving agents obey the rules of the road, as those agents face exactly the same pose, perception, and planning difficulties Skynet does.

Still, the Urban Challenge is more than basic obstacle avoidance. To obey the rules of the road, a vehicle must at some point modify its behaviors according to its and others' absolute locations for queuing at intersections, maintaining safe following distances, and appropriately ignoring cars in oncoming lanes. When the environment is structured, the road constraints provide strong cues to improve Skynet's localization and perception of its surroundings. Team Cornell's scene estimator, described below, identifies and takes advantage of these environmental cues.

3.4.1. Posterior Pose

The scene estimator consists of two algorithms, called "posterior pose" and "track generator," that act as a data pipeline from the localization and perception systems to the planner. The first, posterior pose, is built to take advantage of the position cues hidden in the DARPA road network and the constraints of the Urban Challenge. In particular, it takes advantage of the DARPA-stated assurance that all way points are surveyed accurately, all lanes are marked as indicated in the RNDF, and all stop lines are painted correctly. Under these assumptions, posterior pose uses vision-based lane line and stop line detection algorithms to improve the pose estimator's estimate of Skynet's absolute position. The algorithm, based on GPS map-aiding techniques, sets out to determine a simplified a posteriori joint density of Skynet's pose given road cues:

$$p[E(1:k), N(1:k), \Theta(1:k) | M, Z(1:k)], \tag{3}$$

where $[E(1:k), N(1:k)]$ is Skynet's east–north planar position in the RNDF at time indices 1–k, $\Theta(1:k)$ is Skynet's heading measured counterclockwise from east at time indices 1–k, M is the information obtained from the (static) DARPA RNDF, and $Z(1:k)$ are the available sensor measurements, including output from the pose estimator, two lane-finding algorithms, and a stop line detection algorithm (Miller & Campbell, 2008). In this simplified formulation of the pose estimation problem, Skynet has been constrained from free movement in a three-dimensional environment to planar motion. Note, however, that it still does not assume that Skynet always stays on the road. This design choice simplifies the estimation problem considerably by reducing the size of the pose state vector to 3, compared to 40–50 states in the pose estimator.

The substantial reduction in the size of the pose state vector allows the posterior pose estimation problem to be solved feasibly by a particle filter (Miller & Campbell, 2008). This type of filter was chosen because it is more appropriate for the constrained pose estimation problem than a traditional EKF, as the posterior density in Eq. (3) is often strongly multimodal. In particular, vision-based road cue detection algorithms often commit errors with strong modalities: detecting the wrong lane, detecting two lanes as one, or detecting a shadow as a stop line. The particle filter handles these errors appropriately by maintaining large numbers of hypotheses about vehicle pose, one per particle. Ambiguity about which lane Skynet occupies can then be represented accurately in the distribution of the particles across the road.

Each particle in the posterior pose filter contains one hypothesis of Skynet's three position states: east, north, and heading $[e(k), n(k), \theta(k)]$ within the RNDF. The particle filter is initialized by drawing an initial set of particles according to the approximate posterior Gaussian density implied by the mean and covariance matrix of a single pose estimator packet. From there, particles are predicted forward using differential vehicle motion estimates from the pose estimator. Updates are performed by adjusting the weight on each particle according to how likely its hypothesis about Skynet's pose is correct, as in a traditional bootstrap particle filter (Arulampalam et al., 2002).

The posterior pose particle filter performs one of three types of updates to its particles, depending on which of three sensing subsystems produces a measurement. The first update is an update from the pose estimator, in which the particle filter simply adjusts the weights of its particles based on how closely they agree with the most recent information from the pose estimator. The remaining two updates, lane updates and stop line updates, compare expected road data extracted from the RNDF with local road data measured by three vision algorithms: two lane detection algorithms and a stop line detection algorithm.

Team Cornell utilizes two vision-based lane-finding algorithms to generate measurements of Skynet's distance from the centerline of a lane and heading with respect to the lane. Both of these lane-finding algorithms operate on black-and-white images taken from a Basler A622F mounted on the centerline of Skynet just above the front windshield. One, the MobilEye SeeQ lane-finding software, is available commercially as a lane departure warning system. This system reports position and heading offsets of the lane Skynet currently occupies at approximately 15 Hz. The SeeQ system reliably detects painted lane lines, though it requires several seconds of uninterrupted tracking to become confident in its detections. The other lane-finding algorithm, designed in-house to act as the MobilEye's complement, uses slower but more accurate texture segmentation to find lanes even when roads are not painted (Felzenszwalb & Huttenlocher, 2004). This secondary algorithm produces the same type of lane offset and heading measurements as the MobilEye, though it runs at 2 Hz and starts from scratch at each image frame to avoid creating explicit temporal correlation in its lane estimates. When a measurement from either of these lane-finding algorithms is used to update the posterior pose particle filter, the filter uses the RNDF to compute what each particle expects its position and heading offset to be from the closest lane in the RNDF. Each of these lane hypotheses is then compared with the lane measurement generated by the vision algorithm, and the likelihood of the measurement is used to update particle weights as in the traditional filter update.

Team Cornell also uses a vision-based stop line detection algorithm to generate measurements of Skynet's distance to a stop line. This detection algorithm operates on color images taken from a Basler A311F optical camera mounted in the front grille of Skynet and pointed toward the ground. The algorithm utilizes traditional Canny edge detection to search each image for properly oriented pairs of edges, one a transition from dark road to white stop line paint and its pair a transition from white paint back to road. Distances from the camera are computed for any matches found in this step. These distances are sent to update the posterior pose filter at a rate of 17.5 Hz. When one of these measurements arrives at the posterior pose filter, the filter uses the RNDF to compute the distance between each particle and its closest stop line. These expected distances are then compared with the output of the stop line algorithm, and the corresponding measurement likelihoods are used to update particle weights.

The posterior pose particle filter uses 2,000 particles and runs on a single core of a dual-core Pentium-Mobile system running Windows Server 2003. It processes all measurements from the pose estimator, the lane detection algorithms, and the stop line detection algorithm asynchronously, using a queuing structure similar to the local map to ensure that measurements are applied in the order of their time stamps. The algorithm runs at 100 Hz, the rate at which the pose estimator broadcasts differential vehicle motion estimates. It reports a minimum mean square error (MMSE) posterior pose estimate at 10 Hz to Skynet's data network, along with other metadata, such as a mean square error (MSE) matrix and lane occupancy probabilities. This fused GPS/pose/map/vision posterior pose estimate is used as the sole position feedback signal in Skynet's path planner.

3.4.2. Track Generator

The second scene estimator algorithm, the track generator, combines Skynet's best position estimates from the posterior pose with all the tracked obstacles from the local map to generate high-level obstacle metadata required by the planner. Its primary purpose is to provide a track identification number for each tracked obstacle, one that remains constant over time and allows the planner a means to recognize different intelligent agents over time. This piece of data is one thing that the local map particle-filtered problem framework cannot provide outright, as it reports only the most likely map of tracked obstacles at any point in time: no effort is made within the local map to draw correspondences between similar tracked obstacles residing in separate local map particles.

The track generator poses this track identification problem as an estimation problem, driven by the output of the local map. Because the local map provides a list of all obstacles around Skynet at any point in time, the track generator simply has to match each of those obstacles with its current list of obstacle tracks. Any obstacles that do not match the current list are used to create new tracks. The track generator performs this task using a global maximum likelihood estimator (MLE), implemented in a dynamic programming framework (Cormen, Leiserson, Rivest, & Stein, 2003). First, the likelihood $\lambda_{ij}(k)$ that the ith local map obstacle at time k corresponds to the jth previously identified track is computed for each obstacle/track pair. Three pieces of data are used to compute the correspondence likelihood: bearings of the most clockwise and most counterclockwise points in the obstacle and range to the closest point in the obstacle. The unscented transform is used here as in Section 3.3 to calculate a statistical mean $\mu_i^o(k)$ and covariance $P_i^o(k)$ for the two bearings and the range of the jth obstacle, as well as a mean $\mu_j^t(k)$ and covariance $P_j^t(k)$ for the jth track (Julier & Uhlmann, 1997). From there, the obstacle and track means and covariances are used to compute the likelihood of obstacle/track correspondence as if each were an independent

Gaussian random variable:

$$\lambda_{ij} \sim \mathcal{N}\left[\mu_i^o(k) - \mu_j^t(k),\, P_i^o(k) + P_j^t(k)\right]. \tag{4}$$

The track generator's dynamic programming scheme then searches the sets of correspondences matching each previously existing track to exactly one local map obstacle. The correspondence chosen is the one that maximizes the global likelihood $\Lambda(k)$:

$$\max_{j_1,\ldots,j_n} \Lambda(k) = \max_{j_1,\ldots,j_n} \prod_{i=1}^{n} \lambda_{ij_i}(k) \tag{5}$$

over the correspondences $\{j_1, \ldots, j_n\}$ under the constraint that no two correspondences are the same, i.e., $j_1 \neq j_2 \neq \ldots \neq j_n$. Any old tracks not matched by this scheme are immediately deleted, and any new local map obstacles not matched are assigned fresh track identifications. Each identification number assigned in this manner is unique over the track's lifetime, allowing the planner to reason about an agent by its corresponding identification number.

In addition to the identification number, the track generator estimates four pieces of track metadata that it supplies to the path planner along with the track's state obtained from the local map. The first piece of metadata is lane occupancy: the probability that the track occupies any lanes near it in the RNDF. This probability is calculated via Monte Carlo quadrature. First, all sources of error modeled in estimating the track's location in the RNDF are transformed into a single east–north covariance matrix using a linearized covariance matrix transform similar to that used by the EKF (Bar-Shalom et al., 2001). The east–north covariance matrix is Monte Carlo sampled as a Gaussian random variable, and all sampled particles vote on which lane they occupy to generate lane occupancy probabilities for the track. The second piece of metadata generated for each track is an estimate of whether the track is stopped. This estimate is filtered using a hidden Markov model (HMM) over states "is stopped" and "is not stopped" using local map speeds as a measurement update (Russell & Norvig, 2003). The third piece of metadata is an estimate of whether the obstacle's size indicates that it is carlike. This estimate is also implemented as an HMM but uses the physical size of the track's cluster to provide evidence of whether it is too big or too small to be a car. The final piece of metadata is a track's visibility: whether it is occluded. Track visibility is computed by using the track's cluster points to mask out regions of sensor space under the assumption that the sensors cannot see through any tracks. Any track that is detected as being blocked by another is marked as occluded, and any occluded tracks are allowed to persist in the track generator for up to 1 min without any supporting evidence from the local map. This occlusion reasoning allows the track generator to remember other agents waiting at an intersection that are otherwise blocked from view by a car passing through. It also helps in dense traffic, where other agents may pass into and out of view rapidly.

The track generator reports its list of tracked obstacles and their metadata at 10 Hz on Skynet's data network. It runs on a single core of a dual-core Pentium-Mobile processor, sharing the same process as posterior pose.

3.5. Intelligent Planning

Team Cornell's intelligent planning system uses the scene estimator's probabilistic perception of the environment to plan mission paths within the context of the rule-based road network. The system was designed with three main objectives. First, the planner was designed to be expandable such that additional behaviors could easily be added as the system was developed. Team Cornell used this design requirement to bring portions of the intelligent planner online independently, thus dividing mission planning into more manageable bits of development and testing. Second, the planner was designed to operate asynchronously, so that low-level actuator control loops, middle-level path following, and high-level behavioral planning could all be run at separate rates as computational resources would allow. This design requirement also permitted multiple layers of the planner to be developed in parallel to cope with the aggressive schedule of the Urban Challenge. The final design requirement is stability and robustness: the system should remain stable over time despite unpredictable actions of other intelligent agents.

The intelligent planner developed against these design requirements is split into three primary layers. The top-level behavioral layer combines offline mission information with sensed vehicle and environment information to decide which high-level behavioral state should be executed given Skynet's current context. The middle-level tactical layer then plans a contextually appropriate set of actions to perform given Skynet's current pose and the states of other nearby agents. The low-level operational layer then translates these abstract actions into actuator commands, taking into account road constraints and nearby obstacles. The operational layer also acts as a feedback sensor, verifying whether Skynet achieves the desired behaviors and reporting that completion status to the higher levels of the planner. The planner also consists of several other smaller components, such as the messaging service and communications layer, which facilitate data transfer between the three primary layers of the planner. The following sections describe each of the three primary layers of the planner.

3.5.1. Behavioral Layer

The behavioral layer is the most abstract layer of Team Cornell's planner. Its goal is to decide the fastest route to the next mission checkpoint and then to determine which of four high-level behavior states to use to make progress along that route in the current vehicle state. The first part of that task, route planning, is solved using a modified version of the A* graph search algorithm (Ferguson, Stentz, & Thrun, 2004; Russell & Norvig, 2003). First, the DARPA road network is converted from the RNDF format to a graphical hierarchy of segments defining large contiguous portions of the road, ways defining directions of travel, lanes dividing a direction of travel according to the number of side-by-side vehicles that may be accommodated, and partitions that define a portion of a lane between two GPS way points (Willemsen, Kearney, & Wang, 2003). Zones and intersections are also represented in this graphical framework but as general polygons rather than hierarchical lists of way points. The graph search algorithm then determines the shortest-time path to the next mission checkpoint using dynamically calculated traversal times as costs for road partitions, lane changes, turns, and other required maneuvers. Initial traversal time costs are generated from static information processed from the RNDF and the mission itself, including speed limits, lengths of partitions, lane right-of-way, and the configuration of stop lines at intersections. Dynamic traversal costs, such as road blocks, are also incorporated as large and slowly decaying time penalties on all road partitions adjacent to the location of each blockage as it is discovered.

Once a shortest-time path is planned to the next checkpoint, the behavioral layer repeatedly selects at each planning cycle one of an expandable list of high-level behavior states as most appropriate for making progress along the desired path. The behavioral states used for the Urban Challenge, road, intersection, zone, and blockage, are deliberately defined as broadly as possible to promote stable vehicle behavior through infrequent state changes. Each of these high-level behaviors executes a corresponding tactical component that drives Skynet until the next state change.

3.5.2. Tactical Layer

Each of the four components of the tactical layer is executed when Skynet transitions to its corresponding high-level behavior, as described in Section 3.5.1. All components are similar in that they divide the area surrounding Skynet into mutually exclusive and jointly exhaustive regions. All components also access a common list of intelligent agents currently monitored by the planner, each assigned a region based on the location of its point closest to Skynet. These agents are tracked as a list of independent entities according to their track identification numbers, using state information from the track generator to monitor the agent's speed and shape, what partition the agent occupies, and whether the agent is disabled. A HMM is also run over each agent's partition occupancy probabilities to determine which partitions it most likely occupies in the face of track generator uncertainty (Russell & Norvig, 2003). Differences between the tactical components lie in the types of region monitors they use and in the actions they take in response to events that occur. Each of these is set on a case-by-case basis for each tactical component.

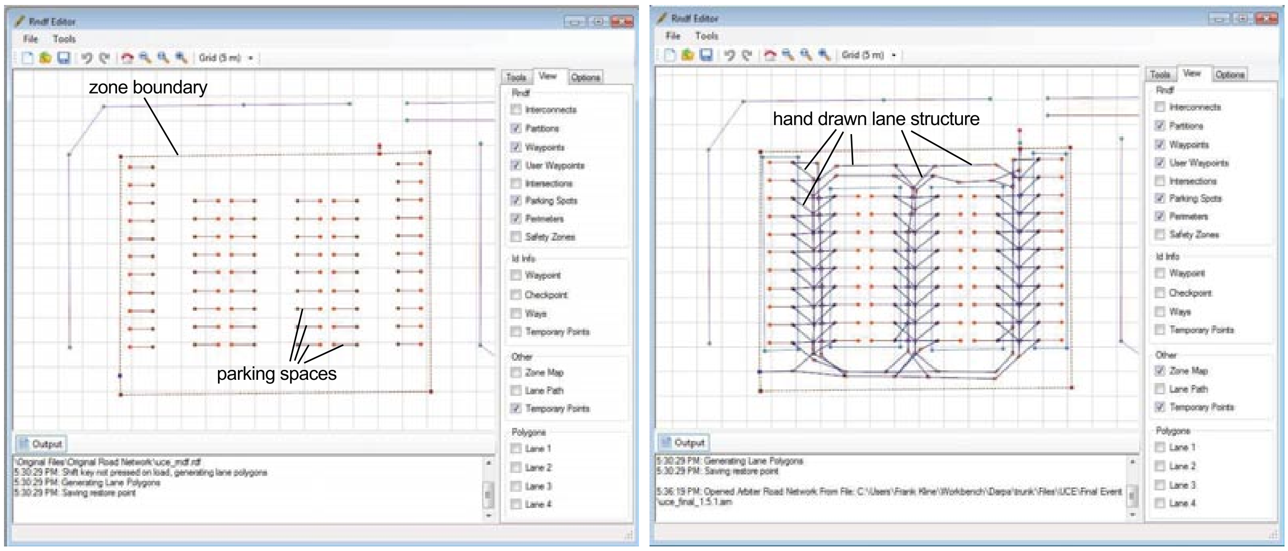
The first tactical component is the road tactical, which controls Skynet when it drives down an unblocked road. This component is responsible for maneuvering into the proper lane to follow the shortest-time path, monitoring other agents in that lane for possible passing maneuvers, and ignoring agents in other lanes when it is safe to do so. Aside from basic lane keeping, these actions are largely performed in response to other agents. The road tactical monitors the agents by dividing the area surrounding Skynet into octants. At each planning cycle, octant monitors check the list of agents to determine how best to proceed along the road. In particular, separate checks are performed in front of Skynet for speed adjustment, adjacent to Skynet for possible lane changes, and behind Skynet for possible closing vehicles and possible reverse maneuvers

(Sukthankar, 1997). The octant monitors are also used as inputs to a decision tree that evaluates the feasibility of a passing maneuver, the only optional maneuver in Skynet's repertoire. The decision tree was developed largely in simulation; it uses heuristic rules based on the distance to Skynet's next mission checkpoint and Skynet's closing speed to slower agents to determine whether Skynet can complete a passing maneuver in the space available. From the decision tree and the octant monitors, the road tactical selects a desired speed and lane for Skynet to follow. This target speed and the lane's local geometry near Skynet are then passed along to the operational layer as a path to track.

The second tactical component is the intersection tactical, which controls Skynet after it achieves an exit or a stop line way point. This component is responsible for achieving proper intersection queuing behavior and safe merging. When executed at an all-stop intersection, the intersection tactical creates monitors for each intersection entry that record agent arrival times by identification number for proper queuing order. The same monitors are set up in merging situations, except that lanes with right-of-way are constantly checked for oncoming vehicles and automatically given priority before Skynet is allowed to merge. When the intersection monitors determine that Skynet is first in the intersection queue, a target speed, a goal point, and a polygon defining the intersection are passed along to the operational layer as a path to track.

The third tactical component is the zone tactical, which controls Skynet after it achieves an entry way point into a zone. This component is responsible for basic navigation in unconstrained zones, including basic obstacle avoidance and initial alignment for parking maneuvers. The zone tactical operates by planning over a human-annotated graph drawn on the zone during RNDF preprocessing. The graph imposes wide artificial lanes and directions of travel onto portions of the zone, allowing Skynet to treat zones as if they were roads. In this way the zone tactical operates much like the road tactical, by traveling along lanes selected by the A* algorithm in the behavioral layer. Figure 6 shows an example lane structure applied to a zone during map preprocessing. Such a lane structure was originally intended to be generated automatically, but time constraints mandated a hand-annotated solution instead.

In parking situations, the zone tactical is simply responsible for navigating into a polygon near the desired parking spot. The obstacle-free polygon,



Rndf: ☐ Interconnects, ✓ Partitions, ✓ Waypoints, ✓ User Waypoints, ☐ Intersections, ✓ Parking Spots, ✓ Perimeters, ☐ Safety Zones; Id Info: ☐ Waypoint, ☐ Checkpoint, ☐ Ways, ☐ Temporary Points; Other: ☐ Zone Map, ☐ Lane Path, ✓ Temporary Points; Polygons: ☐ Lane 1, ☐ Lane 2, ☐ Lane 3, ☐ Lane 4

```
\Original Files\Original Road Network\uce_rndf.rdf
5:30:29 PM: Shift key not pressed on load, generating lane polygons
5:30:29 PM: Generating Lane Polygons
5:30:29 PM: Saving restore point
```

Rndf: ☐ Interconnects, ✓ Partitions, ✓ Waypoints, ✓ User Waypoints, ☐ Intersections, ✓ Parking Spots, ✓ Perimeters, ☐ Safety Zones; Id Info: ☐ Waypoint, ☐ Checkpoint, ☐ Ways, ☐ Temporary Points; Other: ✓ Zone Map, ☐ Lane Path, ✓ Temporary Points; Polygons: ☐ Lane 1, ☐ Lane 2, ☐ Lane 3, ☐ Lane 4

```
5:30:29 PM: Generating Lane Polygons
5:30:29 PM: Saving restore point

5:36:19 PM: Opened Arbiter Road Network From File: C:\Users\Frank Kline\Workbench\Darpa\trunk\Files\UCE\Final Event
\uce_final_1.5.1.arn
```

Figure 6. (Left) A portion of a parking lot zone from the DARPA Urban Challenge RNDF. (Right) The same zone but with Team Cornell's human-drawn lane structure applied during map preprocessing. Skynet uses the lane structure to treat zones like roads, much as a human does when assigning implied directions of travel in a parking lot.

constructed as a workspace for Skynet to use to achieve a desired orientation, is built from sensor data as Skynet approaches the desired parking spot. After Skynet enters the workspace polygon, a cost-based heuristic search method is used to find a series of adjacent arcs to align Skynet to the parking spot. The same algorithm is then used in reverse to pull Skynet out of the parking spot, at which point the zone tactical resumes navigating the imposed lane structure. With the exception of parking, the zone tactical generates the same type of local lane geometry information as the road tactical to send to the operational layer as a path to track.

The final tactical component is the blockage tactical, which controls Skynet when forward progress on the current route is impossible due to obstacle positioning. This component is responsible for detecting road blocks, deciding whether they are temporary traffic jams, and acting accordingly. Team Cornell's blockage detection and recovery relies heavily on the operational layer's constrained nonlinear optimization strategy, described in Section 3.5.3. In particular, the operational layer informs the blockage tactical as to whether there is a forward path through the blockage, what the distance is to the blockage, and whether a reversing or repositioning maneuver is recommended. The operational layer and the blockage tactical then proceed with an escalation scheme to recover from the blockage. First, the blockage is confirmed over multiple planning cycles to ensure that it is not a short-lived tracking mistake. Second, a reversing or rerouting maneuver is executed to find an alternate route on the RNDF, if one is available. If an alternate route is available, the blocked route is given a large time penalty to encourage exploration of alternate routes. The magnitude of the penalty is finite and decays slowly in time, however, to allow Skynet to reexplore the original route if other routes are later discovered to be blocked as well. If no alternate route is available, Skynet's blockage recovery system escalates once more and resets the local map and scene estimator in an attempt to remove any catastrophic mistakes in obstacle detection. If this step fails, planning constraints are relaxed: first the admissible lane boundaries are widened, and then obstacles are progressively ignored in order of increasing size. This process occurs over the course of several minutes without any vehicle progress. This tactical reasoning is unique among the four tactical components, as it relies solely on obstacle avoidance in the operational layer to recover to normal driving.

3.5.3. Operational Layer

The operational layer converts local driving boundaries and a reference speed supplied by the active tactical component into steering, throttle, and brake commands that are used to drive Skynet from one point to the next. The operational layer ultimately has the primary responsibility of avoiding all obstacles when choosing its path, as well as detecting blockages and infeasible paths from the set of obstacles in the commanded driving area. To accomplish this task, the operational layer first processes obstacle estimates from the track generator to prepare them for the trajectory optimizing planner. An obstacle processing loop first transforms obstacle point clouds into polygonal obstacles using a convex hull algorithm (Cormen et al., 2003). Both tracked obstacles and untracked obstacle points undergo this transformation, and metadata are attached to each resulting convex hull to retain all obstacle information sent from the track generator. Required and desired spacing constraints to safely avoid the obstacle are also generated based on the type of object being considered. All obstacle polygons are considered in a vehicle-centric coordinate frame, so they are not affected by GPS error.

Once obstacles are converted to polygons, they are combined with the desired region and speed of travel received to generate a path to follow. First, an initial unsmoothed path is planned through a discretized representation of the obstacles, formed by using nearby obstacle polygons to generate a vehicle-fixed occupancy grid (Martin & Moravec, 1996). The A* search algorithm is then run on the unoccupied portion of this occupancy grid to generate a shortest path through the nearby obstacle field (Russell & Norvig, 2003). The resulting path is never explicitly driven; instead, it is used to determine which obstacles should be avoided on the right of Skynet and which on the left. The initial path is then used to seed a nonlinear trajectory optimization algorithm for smoothing.

The nonlinear trajectory optimization algorithm attempts to smooth the initial base path into a physically drivable path subject to actuator constraints and obstacle avoidance. The algorithm first discretizes the base path into a set of n equally spaced base points $p_i, i \in \{1, n\}$. This discretized base path is formulated such that the path originates from Skynet's current location, and the first base point lies in front of Skynet. A set of n unit-length "search vectors" $u_i, i \in \{1, n\}$

perpendicular to the base path are also created, one for each base point. The trajectory optimizer then attempts to find a set of achievable smoothed path points $z_i = p_i + w_i \cdot u_i, i \in \{1, n\}$ by adjusting search weights $w_i, i \in \{1, n\}$. Target velocities $v_i, i \in \{1, n\}$ are also considered for each point, as well as a set of variables q_i^l and $q_i^r, i \in \{1, n\}$ indicating the distance by which each smoothed path point z_i violates desired spacings on the left and right of Skynet created by the list of polygonal obstacles.

Search weights, velocities, and final obstacle spacings are chosen to minimize the cost function J:

$$J\left(w_i, v_i, q_i^l, q_i^r\right) = \alpha_c \sum_{i=2}^{n-1} c_i^2 + \alpha_d \sum_{i=2}^{n-2} (c_{i+1} - c_i)^2 + \alpha_w \sum_{i=1}^{n} \left(w_i - w_i^t\right)^2 + \alpha_q \sum_{i=1}^{n} \left(q_i^l + q_i^r\right) + \alpha_a \sum_{i=1}^{n-1} a_i^2 - \alpha_v \sum_{i=1}^{n} v_i, \tag{6}$$

where $\alpha_c, \alpha_d, \alpha_w, \alpha_q, \alpha_a$, and α_v are tuning weights; c_i is the approximated curvature at the ith path point; w_i^t is the target search weight at the ith path point; and a_i is the approximated forward vehicle acceleration at the ith path point. Note that the true curvature k_i at each discretized path point is

$$k_i = \frac{2\,(z_{i-1} - z_i) \times (z_{i+1} - z_i)}{\|z_{i-1} - z_i\| \cdot \|z_{i+1} - z_i\| \cdot \|z_{i+1} - z_{i-1}\|}. \tag{7}$$

To simplify differentiation, the following approximate curvature is used instead:

$$c_i = (z_{i-1} - z_i) \times (z_{i+1} - z_i). \tag{8}$$

This approximation is made noting that the initial path points p_i are equally spaced and the search weights w_i are constrained to be small, so the denominator of Eq. (7) is approximately equal for all path points. The approximate forward vehicle acceleration a_i is also calculated under a similar approximation:

$$a_i = \frac{v_{i+1}^2 - v_i^2}{2 \cdot \|p_{i+1} - p_i\|}. \tag{9}$$

The optimized cost function in Eq. (6) has six terms, each corresponding to a particular undesirable driving behavior. The first term penalizes large curvatures, which correspond to undesirably sharp turns. The second term penalizes rapid changes in curvature, which correspond to unstable swerving. The third term penalizes large deviations from the target path offset w_i^t, which force Skynet to move closer to the boundary of its allowed driving region. The fourth term penalizes violations of desired obstacle spacing. The fifth term penalizes sharp accelerations and braking. The sixth term penalizes slow velocities, encouraging faster plan completion time. Heuristic adjustments made to the relative weighting of these penalty terms determines Skynet's driving behavior: whether it prefers aggressive maneuvers or smoother driving.

The cost function $J\left(w_i, v_i, q_i^l, q_i^r\right)$ presented in Eq. (6) is optimized subject to a set of six rigid path constraints:

1. The path must begin at Skynet's current location and heading.
2. Each search weight w_i cannot push the smoothed path outside the boundary polygon supplied by the tactical layer.
3. Each obstacle spacing variable q_i^l and q_i^r cannot exceed any obstacle's minimum spacing requirement.
4. Each true curvature k_i cannot exceed Skynet's maximum turning curvature.
5. Total forward and lateral vehicle acceleration at each path point cannot exceed maximum limits defined by the acceleration ellipse:

$$\left(\frac{a_i}{a_{F,\max}}\right)^2 + \left(\frac{k_i \cdot v_i^2}{a_{L,\max}}\right)^2 \leq 1, \tag{10}$$

   where $a_{F,\max}$ is the maximum allowed forward acceleration and $a_{L,\max}$ is the maximum allowed lateral acceleration.
6. Each search weight w_i and set of slack variables q_i^l and q_i^r must never bring Skynet closer to any obstacle than its minimum allowed spacing.
7. The difference between consecutive path weights w_i and w_{i+1} must not exceed a minimum and maximum.

Additional constraints on initial and final path heading are also occasionally included to restrict the smoothed path to a particular end orientation, such as remaining parallel to a lane or a parking spot.

The constrained optimization problem is solved using LOQO, an off-the-shelf nonlinear nonconvex optimization library. Two optimization passes are made through each base path to reach a final smoothed path. The first step of the smoothed path is then handed to two independent low-level tracking controllers, one for desired speed and one for desired curvature. The speed controller is a proportional-integral (PI) controller with feedback linearization to account for engine and transmission inertia, rolling inertia, wind resistance, and power loss in the torque converter (Centa, 1997; Gillespie, 1992; Wong, 2001). The curvature controller uses an Ackermann steering model with cornering stiffness to convert desired curvature into desired steering wheel angle, which is then passed as a reference signal to a proportional-integral-derivative (PID) steering wheel angle controller (Gillespie, 1992; Wong, 2001). This controller feeds back only on path curvature: lateral offset is not used as a feedback signal because all paths are planned from Skynet's current location and tracked in a Skynet-centric coordinate frame. The optimization is restarted from scratch at each planning cycle and is run at 10 Hz on a dual-core Pentium-Mobile processor running Windows Server 2003.

4. PERFORMANCE AT THE NATIONAL QUALIFYING EVENT

The NQE was held in Victorville, California, from October 25 to 31, 2007, at the Southern California Logistics Airport. At the NQE, the 35 Urban Challenge vehicles invited to participate were tested on three courses, called Area A, Area B, and Area C. Each area, described below in turn, tested one or more specific aspects of autonomous urban driving with focused and carefully monitored scenarios. Only one robot was tested at a time, and any necessary traffic was provided by human-driven vehicles in preset patterns. Safe and sensible driving was paramount in the NQE, with stability and repeatability emphasized in the structure of the event.

4.1. Area A

In Area A, autonomous agents were required to merge into bidirectional traffic flowing at 10 mph (16 km/h) in small concentric loops, extending approximately 100 m in the east/west direction and 50 m in the north/south. Vehicles were required to merge into and out of an unoccupied one-way north/south cross street that bisected the two loops of traffic. Successful merges entered and exited the cross street by completing left-hand turns across oncoming traffic. Traffic was dense, and vehicles were typically given windows of approximately 8–10 s to complete a merge. The lane geometry in Area A was restrictive, with nominal lane widths set at 12 ft (3.66 m) for the inside lane and 10 ft (3.05 m) for the outside lane. Concrete barriers were also placed flush with the boundary of the outside lane, so vehicles could not turn wide while merging. Vehicles were to complete as many of these merges as safely as possible in an allotted time of approximately 30 min.

Team Cornell's vehicle ran in Area A twice, completing 5 successful laps (merges into and out of traffic) in the first attempt at Area A and 10 in the second attempt. In the first attempt, Skynet made two significant mistakes. First, incorrect occlusion reasoning after the second lap led Skynet to conclude that there was an occluded obstacle permanently blocking its turn. The root cause of this was a measurement assignment mistake: a passing vehicle was mistakenly identified as part of a nearby concrete barrier. Team Cornell's vehicle waited for this phantom obstacle for several minutes before being paused and manually reset. This error prompted the team to restrict the lifetime of occluded obstacles to 1 min if they are not supported by sensor data, as indicated in Section 3.4.2.

A different problem occurred on Skynet's fifth lap, where the tactical layer pulled too far to the right of Skynet's lane in response to a nearby oncoming vehicle. During this maneuver, Skynet drove close enough to nearby concrete barriers to violate the operational layer's obstacle spacing constraints. The operational layer reported the path to be infeasible, and Skynet came to an abrupt stop before being paused by DARPA. When allowed to continue, the tactical layer issued a reverse command to Skynet, and the operational layer performed a reverse maneuver to align Skynet in the lane before resuming the mission. No human intervention was required to resolve the error, though the incident drew focus to the fact that the tactical layer could command Skynet to drive to the right of the lane's center without evaluating whether such a command would violate obstacle spacing constraints. This oversight was corrected after the first attempt at Area A and remained in effect for the rest of the Urban Challenge. Minimum and desired obstacle spacings were also permanently adjusted to discourage large evasive maneuvers.

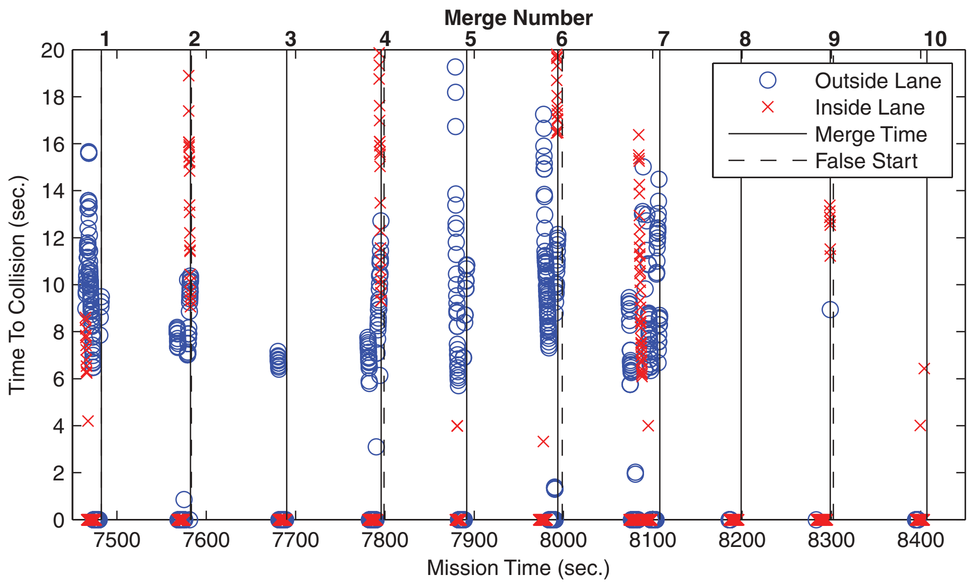


Figure 7. Perceived times to collision with oncoming vehicles in the outside and inside lanes of traffic in Team Cornell's second run of NQE Area A. Solid vertical lines indicate decisions to merge. Dashed vertical lines indicate false starts: times when the planner thinks it is safe to merge and then changes its mind.

Adjustments resulting from the first attempt at Area A made the second attempt more successful. Figure 7 plots the perceived times to collision from oncoming traffic in both lanes in Team Cornell's second run of Area A. Skynet's 10 selected merge times are plotted as solid vertical lines, and the times at which Skynet changed its mind after deciding to merge are plotted as dashed vertical lines. Note that although the local map and track generator often track several cars deep in a line of vehicles, only collision times to the closest vehicles in each lane are plotted. These closest vehicles are those used by the planner to make merge decisions; other tracked vehicles are ignored. Only times at which Skynet is waiting to merge at the stop line are plotted. Times at which the intersection and the 30-m "safety zone" surrounding it are physically occupied by an obstacle are plotted as zero time to collision. Skynet will not attempt a merge when these areas are occupied, unless tracked obstacles in these areas are reliably determined to be stationary. Data are plotted at approximately 10 Hz, once per planning cycle.

Phantom obstacles did not impede vehicle progress in Team Cornell's second attempt at Area A as they did in the first attempt: there are no large periods of time in which the intersection is occupied. Notice that in four of these laps Skynet made false starts, deciding it was safe to merge and then changing its mind. In two of these cases, merges 2 and 6, the tactical layer decided that the oncoming vehicles were threatening enough to postpone the merge. These two cases were subsequently resolved correctly, resulting in successful merges. In the other two, merges 4 and 9, Skynet pulled into the intersection before deciding that the oncoming vehicles posed a threat. As it pulled into the intersection it received more information: improved obstacle speed estimates and therefore more accurate times to collision. In these cases the operational layer could not complete the merge fast enough, and the tactical layer stopped Skynet when it detected approaching obstacles. Neither failed merge was dangerous, though both were undesirable: oncoming traffic stopped, and Skynet continued when it sensed no obstacles approaching. An improved turn reasoning test was added after this test to check whether Skynet had reached a point of no return at the intersection, where it would be more dangerous to stop than to keep going, based on time to collision.

Figure 8 shows a magnified view of collision times in the first merge of Team Cornell's second run in Area A. Each obstacle tracked through the intersection has a smoothly decreasing or increasing time to collision, indicating accurate obstacle speed estimates

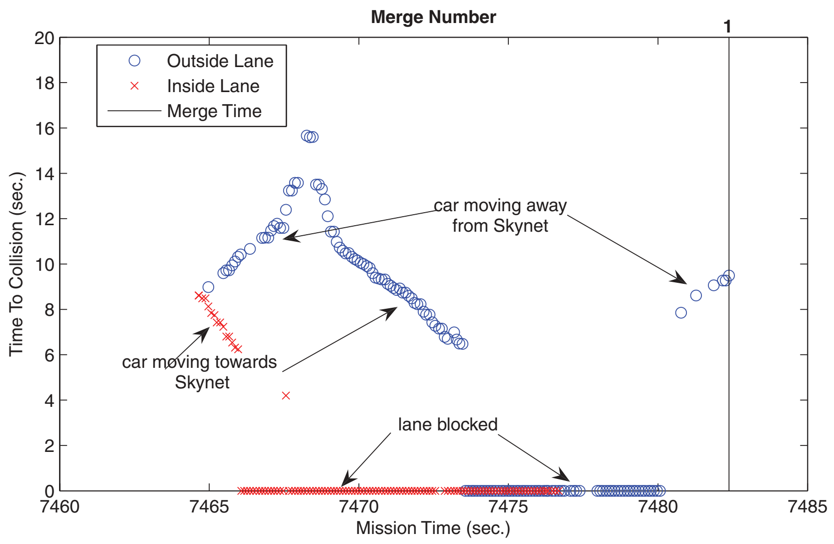


Figure 8. Perceived times to collision in the first merge of Team Cornell's second run of NQE Area A. Here Skynet merges when the nearest obstacle is 10 s away.

provided by the local map and track generator. Occasional increasing collision times reflect uncertainty as to which lane the tracked vehicles occupy. In cases of high uncertainty, the tactical layer conservatively places uncertain vehicles in oncoming lanes and calculates collision times as if they were approaching. These uncertain vehicles are combined with the pool of vehicles with certain lane estimates, and the closest vehicle in each lane is used to evaluate the feasibility of a merge. In the case of Figure 8, the tactical layer attempted a merge when the intersection was free for 10 s, which occurred near time stamp 7482.

4.2. Area B

In Area B, autonomous agents were required to navigate a lengthy route through the urban environment, leaving from a start chute near the grand stands and returning to the mission finish after navigating the course. The environment was devoid of moving traffic but contained a zone with spaces to test parking. A portion of the street was also filled with orange barrels in the center of the street and cars parked at the side of each lane to test obstacle avoidance capabilities. Vehicles also encountered a number of empty intersections and were required to remain in the appropriate lane at all times.

Team Cornell's vehicle ran in Area B twice, failing at the first attempt and succeeding at the second. Skynet failed the first attempt at Area B while trying to park. It successfully aligned itself to the parking space but refused to pull into the space: other unoccupied cars parked nearby violated admissible obstacle spacing constraints in the operational layer. After sitting for several minutes, Skynet was manually stopped, positioned inside the parking space, and allowed to continue. Skynet proceeded as normal, entering the portion of the course testing obstacle avoidance. There, Skynet passed several parked cars and orange barrels, until coming upon a section of road with parked cars at the outer sides of both lanes. Believing both lanes to be blocked, Skynet began a U-turn maneuver but ran out of time before completing the course. All behavior until the parking difficulty was normal, and Skynet navigated without incident.

Behavior in the first attempt at Area B prompted the team to make adjustments to spacing constraints in the optimization problem presented in Section 3.5.3 and to the tactical layer. In particular, minimum spacing constraints in the tactical layer were reduced from 0.9 to 0.3 m for stopped obstacles and from 1.4 to 0.9 m for moving obstacles. These adjustments allowed Skynet to park in parking spaces closely surrounded by obstacles. They also made Skynet less sensitive to concrete barriers and other stationary obstacles at the sides of the lanes. Figure 9 (top) plots the sensed spacing between Skynet and other obstacles in a portion of the Area B course lined with parked

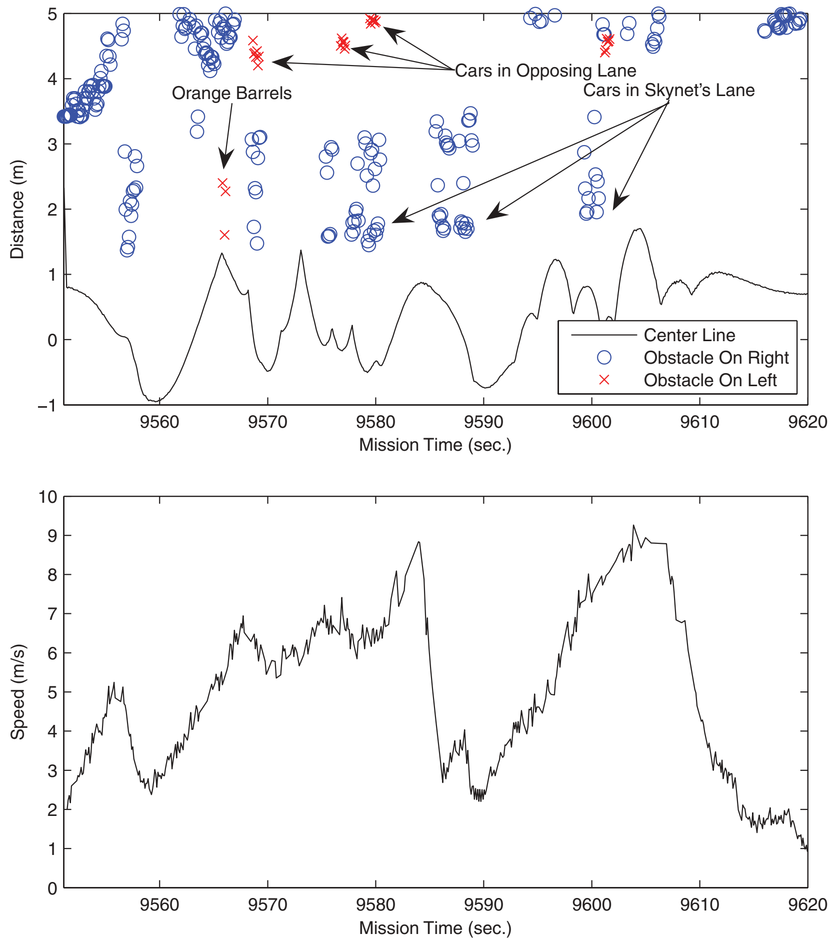


Figure 9. (Top) Spacing between Team Cornell's vehicle and stationary obstacles during a portion of the second run of NQE Area B. (Bottom) Vehicle speed through the same portion of Area B.

cars: 10 in Skynet's lane, 4 in the opposing lane, and 1 group of 4 orange barrels clustered in the center of the road. The lane width in this portion of the course was marked in the RNDF as 12 ft (3.6576 m). Changes to spacing constraints after the first attempt at Area B allowed Skynet to navigate this portion of the course without stopping, as shown in Figure 9 (bottom). Notice that to achieve these spacings, however, Skynet at times crossed the center line of the road by up to 1 m. Such behavior is heavily penalized at the operational layer and occurred only when the obstacle constraints mandated it.

Navigation in the remainder of Area B proceeded without incident. Skynet did, however, receive several spurious disabling commands over the DARPA-provided emergency stop wireless interface. The source of these commands is still unknown, though DARPA technical staff concluded that they came from an external source. Skynet was allowed to be restarted after the first of these spurious signals, though it was eventually removed from the course as more emergency stop signals were received. When removed, Skynet was approximately 60 m from the final checkpoint of the mission.

4.3. Area C

In Area C, autonomous agents were required to navigate a short loop with two four-way intersections. The intersections were populated with an increasing number of traffic vehicles to test various configurations of intersection queuing. If a vehicle successfully

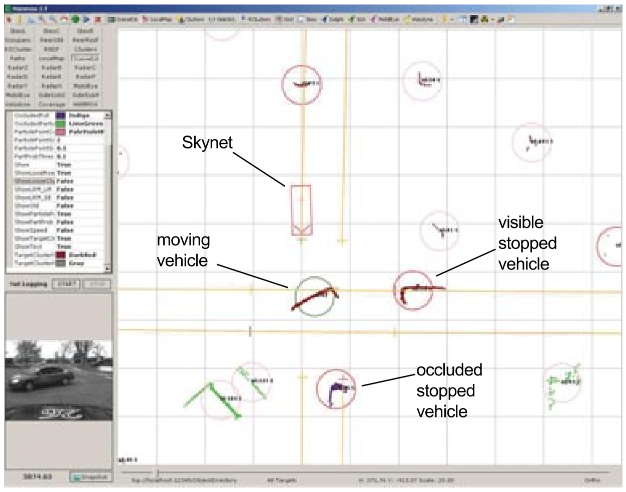


Figure 10. Occlusion reasoning at a sample intersection configuration in NQE Area C allows Skynet to remember vehicles at the intersection even when blocked from view. Here Skynet waits at the north entrance to the intersection, the topmost in the screen.

negotiated all intersection scenarios correctly, road blocks were placed across the road to force the vehicle to plan a new route to complete its mission. These road blocks were left in place for the remainder of the mission to ensure that vehicles remembered previously encountered road blocks when planning new routes.

Team Cornell's vehicle ran in Area C only once, completing the entire course without incident. Figure 10 shows output of the posterior pose and track generator algorithms at one of the most difficult intersection scenarios. In this scenario, human-driven vehicles populated each entrance to the intersection. Each of these vehicles had precedence over Skynet and, when turning, passed in front of Skynet to block the remaining vehicles from view. This scenario exercised Skynet's occlusion reasoning, which was invoked each time one vehicle passed in front of the others. One example is given in Figure 10, which shows Skynet waiting at the northern entrance to the intersection. A second vehicle, which has precedence over Skynet, entered the intersection from the west. As it passed in front of the vehicle sitting at the southern entrance, occlusion reasoning marked that vehicle as occluded. The planner held the occluded vehicle's place in the intersection queue, and the vehicle was tracked correctly through the entire intersection scenario. The vehicle waiting at the eastern entrance was similarly marked as occluded when it was blocked from view and successfully tracked through the scenario as well.

In completing Area C, Team Cornell's vehicle correctly solved seven intersections: one empty, one with one other vehicle, three with two other vehicles, one with three other vehicles, and one with four other vehicles. Of these scenarios, three resulted in occlusions; each occlusion was correctly handled by obstacle tracking and occlusion reasoning in the local map and track generator, respectively.

After the intersection scenarios, Skynet was forced to plan around road blocks. Figure 11 shows the different steps in Skynet's reasoning while handling these road blocks. First, Figure 11 (top) shows Skynet traveling along its initial path, the shortest path to the next mission checkpoint. This path took Skynet down the southern route of Area C, where it encountered the first road block (eight orange barrels blocking both lanes), shown in Figure 11 (middle left). Figure 11 (middle right) shows Skynet's path around the blockage, which took it through the central route of Area C. Halfway down that new path, Skynet discovered a second road block (a horizontal metal bar with six stop signs blocking both lanes), shown in Figure 11 (bottom left). Figure 11 (bottom right) shows the path planned around the second road block, which took Skynet around the northern path of Area C. In planning this third path, Skynet remembered the locations of both the road blocks it had previously encountered. These blockages heavily penalized the central and southern routes of Area C, as described in Section 3.5.1. This made the northernmost path least expensive, even though it was the longest.

5. PERFORMANCE AT THE URBAN CHALLENGE EVENT

The Urban Challenge Event (UCE) was held in Victorville, California, on November 3, 2007. Of the 35 teams originally invited to participate in the NQE, only 11 were invited to continue on to the UCE. The UCE course, shown in Figure 12, subsumed NQE Areas A and B and roads connecting them. Most roads were paved with a single lane in each direction, as they belonged to a former residential development. Several roads admitted two lanes of traffic in each direction. One road, in the southeastern corner of the network, was a dirt road that descended a steep gradient.

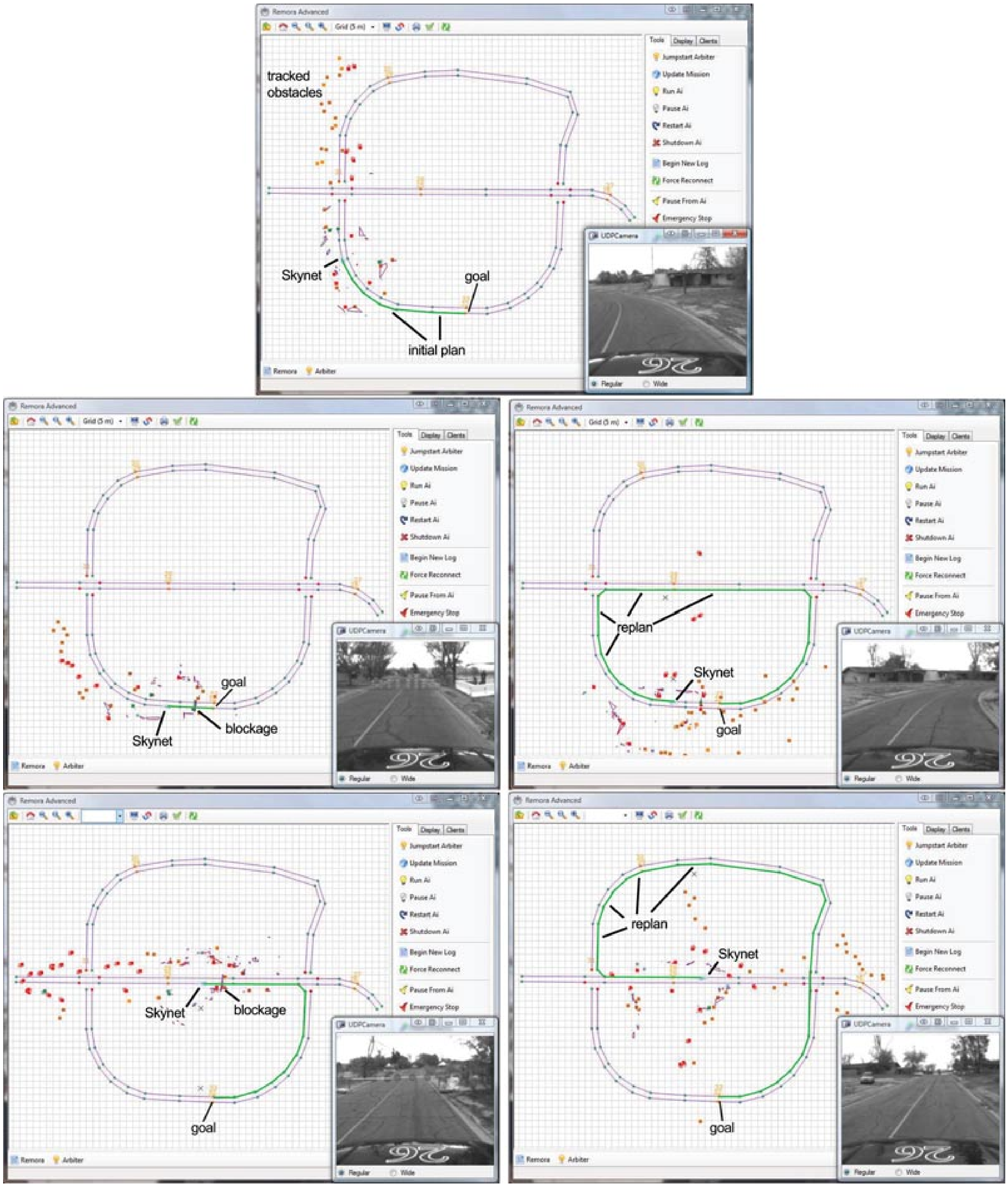


Figure 11. Blockage recovery behaviors in NQE Area C. (Top) The initial planned path. (Middle left) A road block forces Skynet to plan a new path. (Middle right) Skynet follows its new plan. (Bottom left) Skynet encounters a second blockage. (Bottom right) Skynet plans a third path, remembering the positions of both blockages.

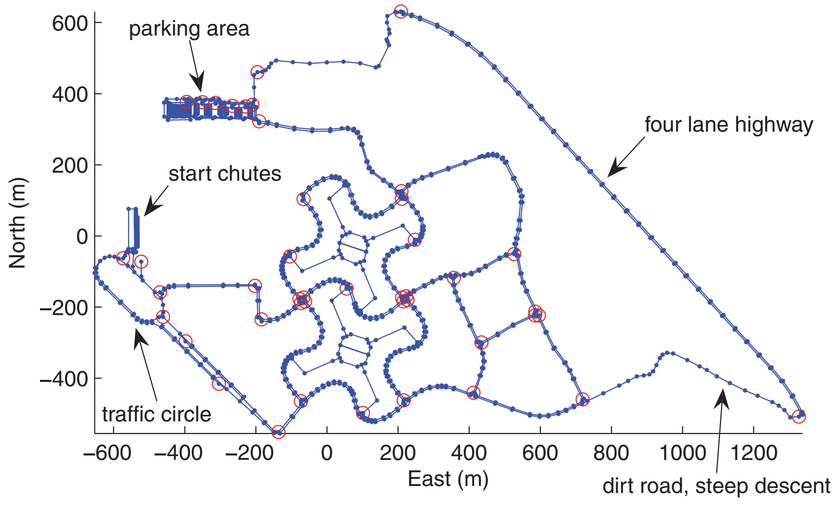


Figure 12. The Urban Challenge road network. Way points are represented by small dark dots, with traversable lanes and zone boundaries represented as lines connecting them. Stop lines are represented by large circles.

Each vehicle competing in the UCE was required to complete three missions, defined by separate MDF. Each mission began among an array of start chutes in the western end of the road network. The first two missions given to Team Cornell were each subdivided into six "submissions," which required Skynet to achieve a series of checkpoints before returning to perform a loop in the traffic circle at the western end of the road network. The final mission for Team Cornell was subdivided into seven such submissions. Each mission ended with Skynet returning to a finishing area near the start chutes, where the team could recover Skynet and reposition it for the next mission. The missions given to Team Cornell totaled approximately 55 miles (88.5 km) of driving on the shortest path. All 11 qualifying robots were allowed to interact in the UCE course simultaneously, with additional traffic supplied by human-driven Ford Tauruses. As in the NQE, safety and sensible behavior were paramount, though no official judging criteria or scores were made public for the UCE.

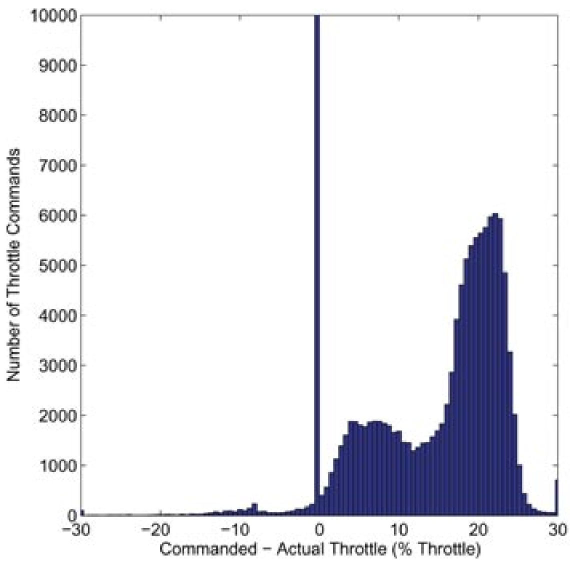
Team Cornell's performance in the UCE is presented in three sections. Section 5.1 gives Skynet's general behavior and performance in the UCE. Section 5.2 presents a small case study of another unique event, when Skynet encountered a human-driven Ford Taurus driving the wrong way on a one-way road. A second event, when Skynet correctly tracked two slow-moving vehicles and passed them, is presented in Section 5.3. Section 5.4 presents a small case study of a third unique event occurring near the corner of Washington and Utah Streets, where Team Cornell's Skynet waited for approximately 12 min on the wrong side of the road. A fourth event, a minor collision with the MIT robot, is left as the topic of a companion paper.

Six animations, three from the NQE corresponding to Areas A, B, C, and three from the UCE corresponding to Sections 5.2–5.4, are available in the online issue at www.interscience.wiley.com.

5.1. Overall UCE Performance

Team Cornell's Skynet was one of only six vehicles to complete the Urban Challenge; the other five vehicles were disqualified at various points in their respective first missions. Skynet completed the Urban Challenge in 5 h, 55 min, 1 s, with a total of approximately 55 miles (88.5 km) of autonomous driving.

Although Skynet's overall performance in the UCE was successful, several poor behaviors were displayed repeatedly. First, a low-level electrical grounding problem in the throttle actuation system developed over the course of the UCE, forcing Skynet to travel much slower in the final mission of the UCE than in the first two. Figure 13 (left) shows evidence of the problem in the final mission of the UCE. In the final mission, 33.9% of all commands received at



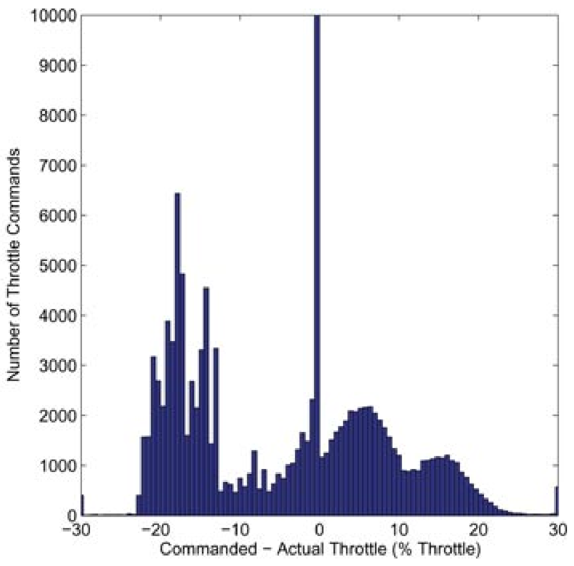


Figure 13. (Left) Histogram of the difference between commanded and received throttle positions in the final mission of the Urban Challenge. A low-level electrical grounding problem resulted in Skynet achieving significantly lower speeds in the final mission of the Urban Challenge. (Right) A similar histogram, from the first mission of the Urban Challenge, shows no evidence of the problem.

Skynet's engine control unit (ECU) exactly matched those issued by the operational layer. In contrast, 64.4% of the commands issued were larger than those received at the ECU, and only 1.7% were smaller. Figure 13 (right) shows the same plot for the first mission of the UCE: only 24.5% of the commands were larger than those received, 36.42% were smaller, and 39% were equal. The obvious disparity in commanded and received throttle positions in the last mission prevented Skynet from achieving its commanded speed most of the time, resulting in significantly slower progress in the final mission of the UCE. The problem is most apparent in mission completion times: Team Cornell spent 1 h, 51 min, 39 s on the first mission, 1 h, 18 min, 40 s on the second, and 2 h, 45 min, 12 s on the third. In simulations without any traffic, Skynet finished the missions in 1 h, 15 min; 1 h, 17 min; and 1 h, 40 min, respectively. Note the timing in the second mission: Skynet completed the mission only 2 min slower than it could in simulation. The disparity in the completion times of the first mission, due to the Washington and Utah event, is explained in Section 5.4.

Ultimately, Skynet's low-level grounding problem is owed to an oversight in circuit design at the actuation level. Had Team Cornell opted for a commercial actuation package, the problem might have been avoided. However, it is unfair to conclude that a commercial package is superior based on this one issue, as commercial options would have undoubtedly presented their own unforseeable growing pains and implementation drawbacks. With an in-house implementation, the team was at least able to modify the actuation scheme as the rest of the system matured. And, all considerations of the Urban Challenge aside, the in-house scheme gave team members a beneficial learning opportunity.

In addition to low-level-speed problems, Skynet also occasionally commanded fast stops as a result of perception and tracking errors. Such fast stops, defined as emergency zero-speed commands issued from the tactical layer, occurred 53 times during the UCE. Of these, 24 occurred in the first mission, 17 in the second, and 12 in the third. Because not all of these fast stops are perception errors, it is instructive to look at the location of Skynet at the times of these emergency commands. These locations are represented as black squares in Figure 14 (left). Most of the emergency commands issued by the tactical layer occurred in the southwestern portion of the UCE road network, in areas where concrete barriers lined the wall of the course. These concrete barriers presented significant challenges to the clustering, tracking, and path planning algorithms. The full extent of the barriers was often not visible, so the clustering

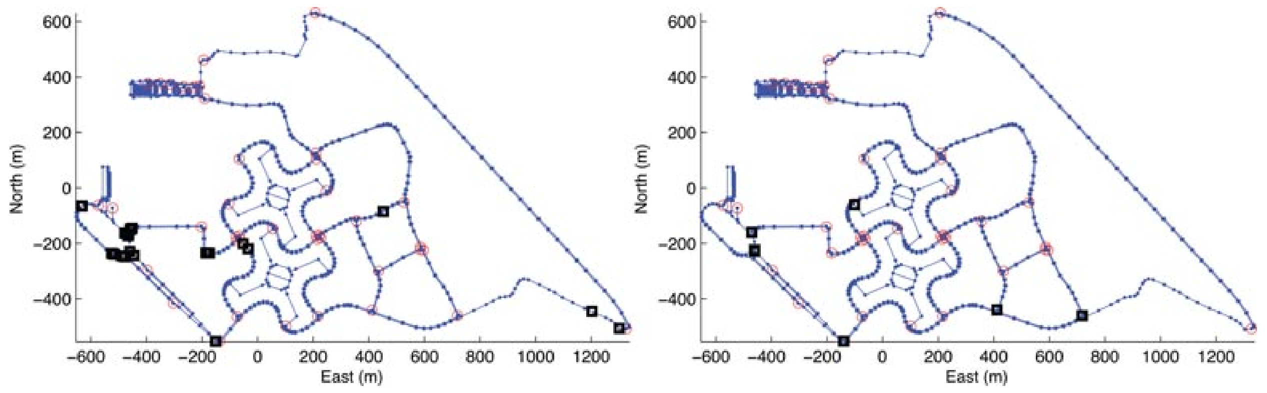


Figure 14. (Left) Location of Skynet (black square) during the 53 emergency brake slams it performed during the Urban Challenge. (Right) Location of Skynet (black square) during the 10 times it went into blockage recovery during the Urban Challenge.

algorithm occasionally clustered the concrete barriers with other obstacles in the lane. Furthermore, measurement assignment mistakes occasionally caused measurements of concrete barriers to be pulled into the lane and applied to other vehicles, causing phantom obstacles to appear in the local map. Finally, the concrete barriers were occasionally close enough to the lane boundary that the operational layer would report a path to be infeasible. The latter problem was most catastrophic, as it caused Skynet to think that the road was blocked.

Figure 14 (right) plots the location of Skynet at each of the 10 times the blockage recovery tactical component was executed: once in the first mission, six times in the second, and three times in the third. These were the worst behaviors displayed by Skynet, resulting in reverse maneuvers or basic unconstrained obstacle avoidance. Those in the western part of the course were caused by the position of the concrete barriers relative to the DARPA-supplied RNDF way points. The remainder were generally caused by perception mistakes: either incorrect clustering or incorrect tracking.

Despite the throttle problem and the emergency commands, Skynet's overall performance was still safe and reliable enough to finish the UCE. The local map and track generator tracked a total of 175,252 distinct obstacles during the course of the UCE, with only 53 causing mistakes in Skynet's behavior. The average tracking lifetime of the obstacles was 6.8 s. Of these, 26.5% were estimated as being on the RNDF at some point in their lifetime; these obstacles had an average tracking lifetime of 10.7 s. Track identification numbers were also very stable during the race, with 13,609 tracks lasting for more than 15 s. On average, the local map and track generator maintained 48.5 tracked obstacles at each iteration, with a maximum of 209 obstacles tracked in one iteration of the algorithms. The intelligent planner was also similarly stable, despite the 10 mistaken blockage recovery actions. The system operated in the road tactical component 76.1% of the time, in the intersection tactical component 20.8% of the time, in the zone tactical component 2.7% of the time, and in the blockage tactical component 0.5% of the time.

5.2. The Wrong-Way Vehicle

One unique event that happened to Skynet over the course of the UCE occurred early in the first mission, when Skynet encountered and properly dealt with a moving vehicle traveling the wrong way on a one-way road. Skynet encountered this vehicle, a human-driven Ford Taurus, on the dirt road at approximately (1,202E, −446N) in the coordinate frame of Figure 12. Figure 15 (top) shows the oncoming Taurus from the point of view of the optical cameras and the operational layer. Note that the Taurus pulled to the left of the lane as far as possible to minimize the chance of collision.

As the wrong-way vehicle moved closer to Skynet, the road tactical component largely ignored it due to assumptions made in the road tactical component. The road tactical component's forward vehicle monitor assumes that all vehicles move in their lanes

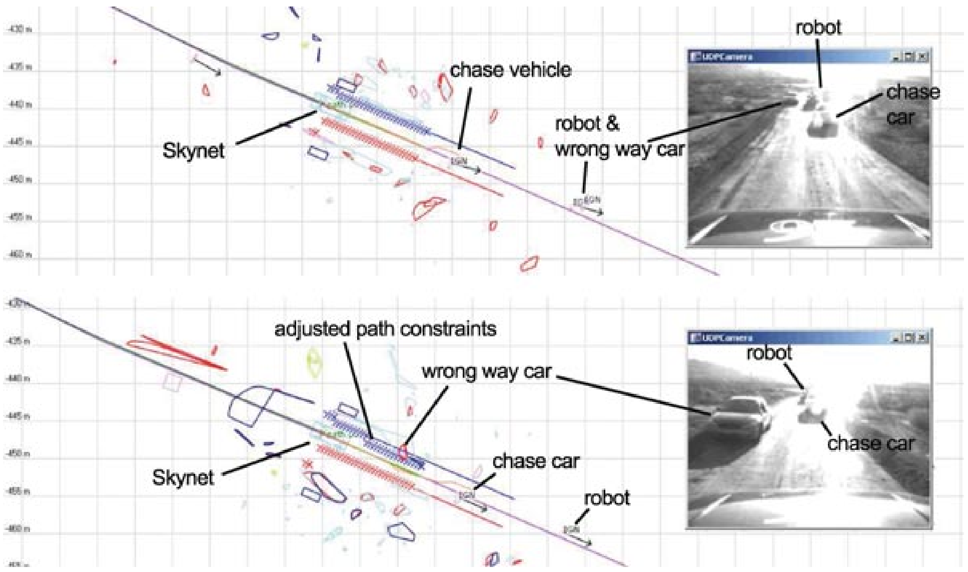


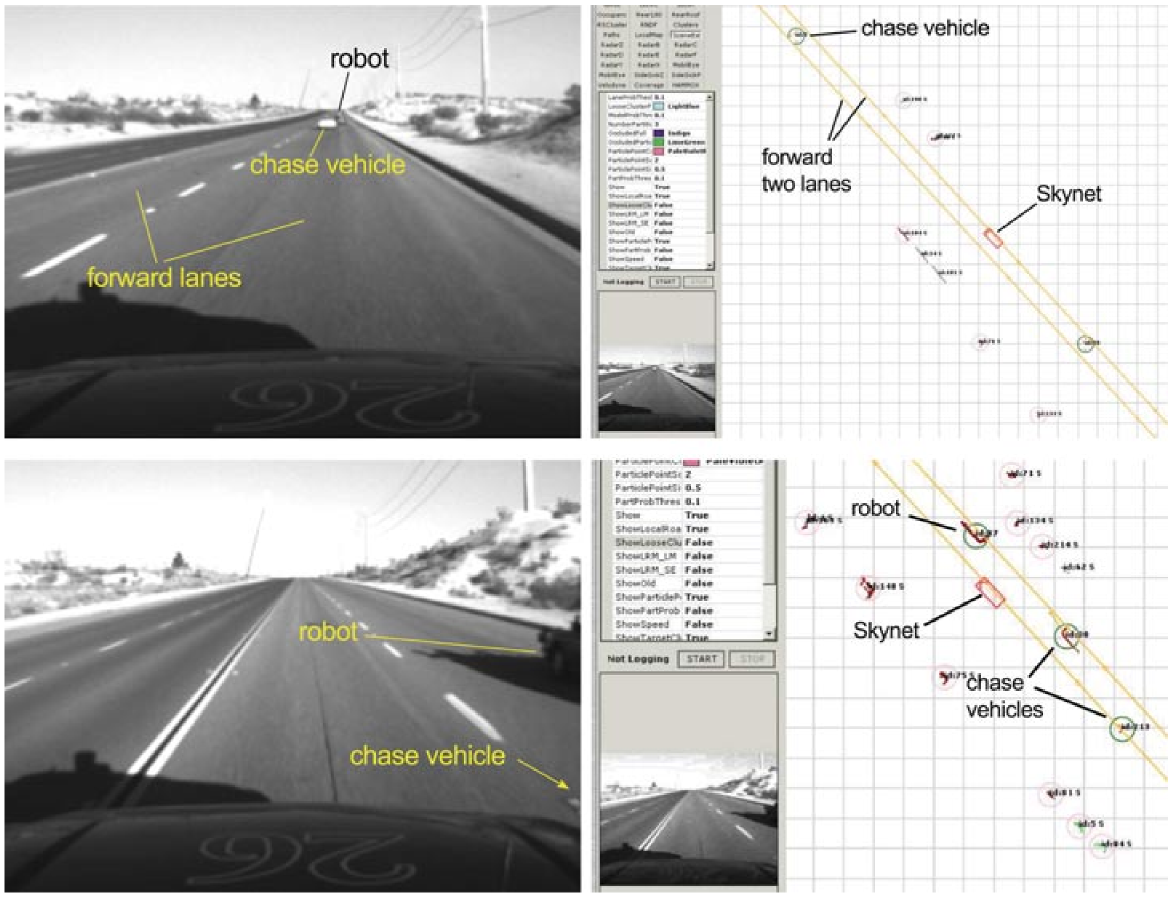
Figure 15. (Top) Team Cornell encounters a human-driven vehicle traveling the wrong way on a one-way dirt road. (Bottom) The wrong-way vehicle is avoided by the operational layer, which adjusts path constraints as if the vehicle were a static obstacle.

in the proper direction; therefore, only the closest vehicle in front of Skynet is monitored. As a result, the wrong-way vehicle was entirely ignored until it was the closest obstacle to Skynet. At that point the road tactical component began to monitor the vehicle, and it would have commanded a fast stop if the vehicle continued to move. Instead, the wrong-way vehicle stopped moving. The clustering algorithm subsequently grouped it with nearby bushes and the dirt berm. As a result, the operational layer avoided the wrong-way car as a static obstacle, as shown in Figure 15 (bottom). After successfully avoiding the car, Skynet continued with its mission. This unique incident is one that highlights the capabilities of the local map and constraint-based planner to handle unforeseen circumstances: had these systems made more brittle assumptions about what they were tracking or avoiding, this situation could have easily resulted in a collision.

5.3. The Pass

A second unique event that Skynet encountered in the UCE was a successful high-speed pass of a slower moving robot and a human-driven Ford Taurus following it. This event occurred in the first mission, at approximately (947E, −40N) in the coordinate frame of Figure 12, shortly after Skynet encountered the wrong-way vehicle. As Skynet traveled along in its lane, it encountered the slow-moving vehicle and trailing human-driven Ford Taurus. Figure 16 shows the encounter.

Vehicles in the desired lane of travel must satisfy two conditions for the road tactical component to execute a passing maneuver. First, the vehicle must be classified as slow moving by the forward vehicle monitor of the road tactical component. This is defined by satisfying three criteria: the vehicle must be tracked consistently for at least 3 s, the vehicle must not be in an intersection safety zone, the vehicle must be traveling slower than Skynet, and the vehicle must be traveling at less than 70% of the maximum speed allowed on that particular road. Second, the road tactical component must determine that there is enough time to pass the vehicle, a calculation based on the distance remaining before the next mission checkpoint and the difference in speed between the tracked vehicle and Skynet. In this situation, both these conditions were met. Skynet correctly tracked the trailing vehicle at 118 m, determining its speed to be 7.1 m/s compared to Skynet's 13.7 m/s. The vehicle was considered slow moving at a time



```
ParticlePointSi 2
ParticlePointSi 0.5
PartProbThres   0.1
Show            True
ShowLocalRoa    True
ShowLooseClu    False
ShowLRM_LM      False
ShowLRM_SE      False
ShowOld         False
ShowParticleP   True
ShowPartProb    False
ShowSpeed       False
ShowTargetCl    True
```

Figure 16. (Top left) Team Cornell approaches two slow-moving vehicles. (Top right) The tracked vehicles are determined to be slow moving, and Skynet decides a passing maneuver is feasible. (Bottom left) Skynet passes both vehicles at 30 mph (48.3 km/h). (Bottom right) Skynet tracks both vehicles as it passes them.

when the next checkpoint was 1,070 m away, which was sufficient time to complete a passing maneuver. Figure 16 (top left) shows the vehicles' configuration when Skynet decided to pass. Figure 16 (top right) shows the track generator tracking the trailing vehicle just prior to beginning the pass.

After checking for other fast-moving vehicles approaching from the rear and other slow-moving vehicles in the adjacent lane, Skynet deemed the passing maneuver to be safe. As Skynet moved to the adjacent lane, the forward vehicle dropped from Skynet's front radar detection cone. This caused a measurement assignment mistake, and a second duplicate track was created near the original vehicle. This duplicate appeared near the lane boundary, so Skynet slowed to match its speed. The local map and track generator resolved the mistake after several seconds, allowing Skynet to speed up to pass.

As Skynet passed the first slow-moving vehicle, shown in Figure 16 (bottom left), it also tracked the slow-moving robot in front. Figure 16 (bottom right) shows the output of the track generator at one iteration, with tracks assigned to both the slow-moving robot being passed by Skynet and to the slow-moving Taurus that was just passed. Unable to change back into the right lane while the slow-moving robot occupied it, Skynet continued to drive in the passing lane. Once clear of both slow-moving vehicles, Skynet sensed no obstructions in the right lane, either in front or to its side. Skynet therefore pulled back into the right lane, to complete the passing maneuver and continue with its mission.

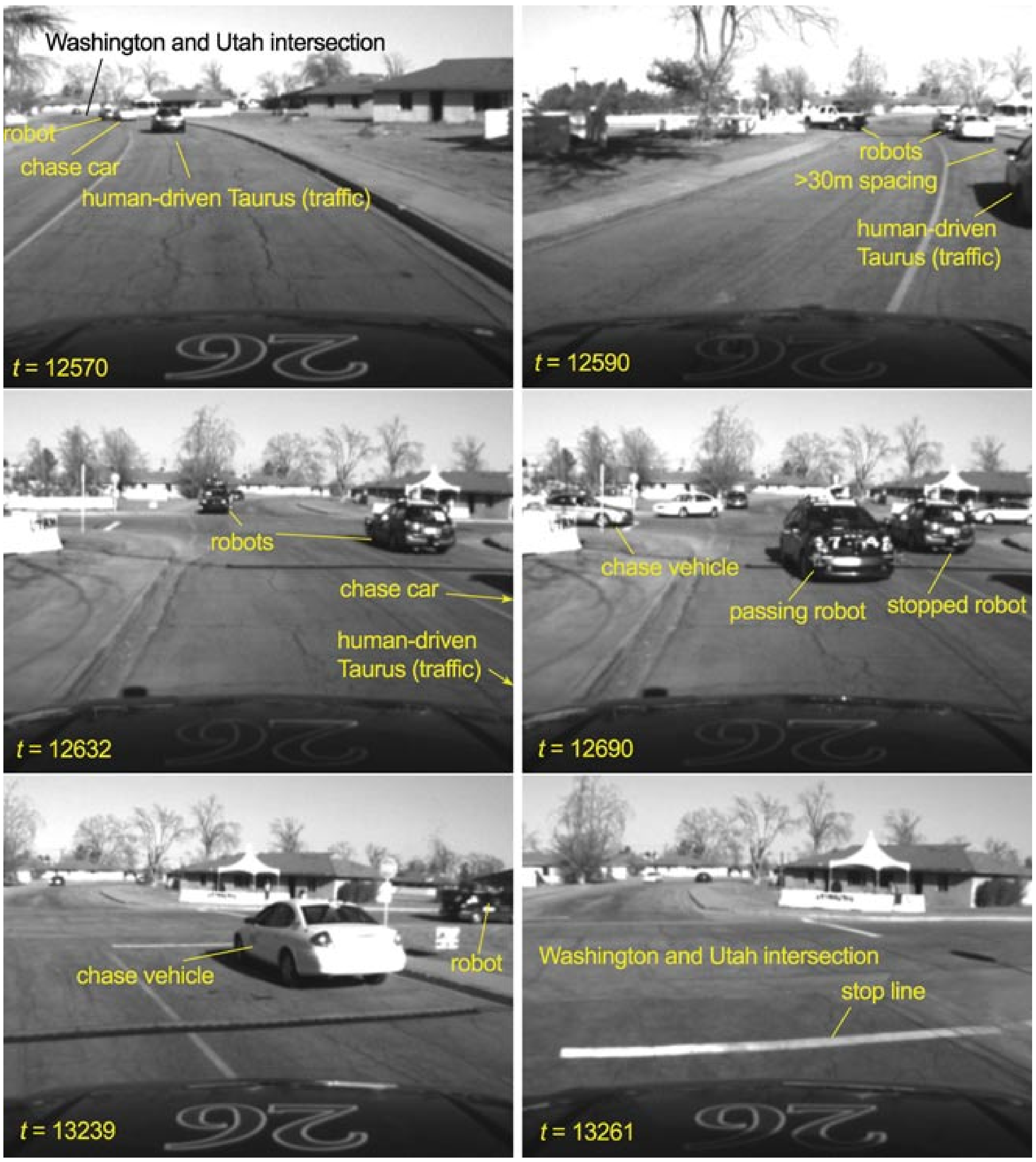


Figure 17. (Top left) Skynet approaches the traffic jam at the corner of Washington and Utah. (Top right) Skynet begins to pass a stopped vehicle deemed disabled. (Middle left) Skynet continues its pass but gets trapped in the wrong lane as its gap closes. (Middle right) Other vehicles pass Skynet as it waits for the intersection to clear. (Bottom left) The traffic jam clears. (Bottom right) Skynet pulls to the stop line to continue the mission.

The passing maneuver executed by Skynet illustrates the complex dependencies among all its constituent subsystems. The actuation provided real-time feedback necessary to complete the maneuver smoothly. The pose estimator and posterior pose produced a smooth and reliable solution that allowed precision path tracking. The local map and track generator tracked all the slow-moving vehicles with consistent speed estimates, even as the vehicles passed from one sensor to another. The road tactical utilized all information from the sensing to reason about whether a pass could be completed successfully, and the operational layer executed the desired maneuver. The passing event was one significant instance among many in which all components of Skynet functioned together correctly and is an accurate representation of Skynet's performance in the UCE.

5.4. The Traffic Jam at Washington and Utah

A final unique event for Team Cornell occurred during a traffic jam near the corner of Washington and Utah streets at approximately (−41E, −208N) in the coordinate frame of Figure 12. As Skynet approached the intersection, it encountered a traffic jam: a robot was disabled at the stop line. In line behind the vehicle were two human-driven Ford Tauruses, one following the robot, the other acting as traffic. Skynet pulled to a stop behind the last of these three vehicles, as shown in Figure 17 (top left).

The last of the vehicles, a human-driven Taurus, left a large space between it and the other two vehicles. Unfortunately, this large space was farther than the 30-m DARPA-specified safety zone that surrounds each intersection. Because the vehicle was outside the safety zone, Skynet was allowed to consider it disabled and pass. After waiting 10 s, Skynet did exactly that: concluded that the Taurus was disabled and began to pass it.

Shortly after Skynet began to pass, it observed another robot turning into the oncoming lane, as shown in Figure 17 (top right). The planner immediately executed an emergency brake command, one of the 53 mentioned in Section 5.1. The planner prevented Skynet from moving while the oncoming vehicle passed, and a collision was avoided. A human-driven Taurus following the oncoming robot also turned into Skynet's lane, but immediately drove onto the sidewalk in order to avoid potential collisions. With the lane clear, Skynet continued its passing maneuver of the original Taurus, which still had not moved. As Skynet pulled fully into the opposing lane, however, the Taurus closed the gap, preventing Skynet from returning to the proper lane to complete the pass.

With nowhere left to go, Skynet concluded that its desired lane was full and began to wait, as shown in Figure 17 (middle left). While waiting, Skynet invoked the blockage recovery tactical component, one of the 10 instances mentioned in Section 5.1. Although the recovery process escalated several levels, including resetting all layers of the planner, Skynet retained the correct view of the situation. During this period a number of other vehicles also passed by without incident, as shown in Figure 17 (middle right). Although Skynet had ample opportunity to navigate the intersection as an unstructured zone, conservative design choices deliberately prevented Skynet from adopting a pure obstacle avoidance strategy so close to an intersection for safety reasons.

Approximately 12 min later, the disabled robot resumed its mission, as shown in Figure 17 (bottom left). As soon as the intersection cleared, Skynet pulled into the proper lane, stopped at the stop line, and continued with its mission, as shown in Figure 17 (bottom right). The event at Washington and Utah is unique in the Urban Challenge because no rules were broken: robots were allowed to pass other disabled robots after waiting for 10 s, provided that they were outside intersection safety zones. Despite following the rules, Skynet still performed undesirably. Such a situation emphasizes the importance of higher level reasoning about other vehicles' behavior that was not incorporated into the Urban Challenge: had Skynet understood the root cause of the traffic jam, it would never have tried to pass in the first place.

6. CONCLUSIONS

This paper has presented a high-level system-by-system view of Team Cornell's Skynet, one of six vehicles to successfully complete the 2007 DARPA Urban Challenge. Skynet consists of several sophisticated subsystems, including in-house actuation, power, and networking design, a tightly coupled attitude and pose estimator, a multitarget obstacle detection and tracking system fusing multiple sensing modalities, a system to augment the pose solution with computer vision algorithms, an optimization-based local path planner, and a state-based urban reasoning agent. The vehicle built from these components can solve complex merging and intersection scenarios, navigate unstructured obstacle fields, park in designated spaces, and obey basic traffic laws.

The success of Team Cornell's Skynet was largely based on lessons learned in the 2005 Grand Challenge (Miller et al., 2006). In particular, the team opted for standard automotive components, while still designing the actuation and power distribution systems in-house. In addition, the team learned to be wary of GPS, designing a more robust tightly coupled pose estimator than the loosely coupled version used in the Grand Challenge. Mistrust of GPS signals also caused the team to build both an obstacle tracking system and an obstacle avoidance system that operated in a vehicle-fixed coordinate frame, completely independent of GPS. The instability of greedy search spline-based planners motivated Team Cornell to adopt a local path planner that selected paths based on physical constraints in the real world, including actuator constraints, obstacle avoidance constraints, and preferences for smoother, straighter paths. These lessons led to Team Cornell's success in the Urban Challenge.

Though successful, Team Cornell exposed several critical areas for continued investigation. In particular, modeling and estimation of other vehicles' behavior would have vastly improved Skynet's performance in the Washington and Utah traffic jam and would have allowed more sophisticated reasoning to take place elsewhere. Further modeling of the environment itself, including a GPS-independent local road model, would also aid in Skynet's situational awareness in cases in which the road is poorly surveyed. Finally, the Urban Challenge has exposed the need for a better understanding of the interplay between probabilistic sensor information and robust planning. The state-based planner used on Skynet was stable, but only after many months of careful tuning. Each of these research areas remains largely unexplored, though progress will be necessary to permit improvements in autonomous driving.

ACKNOWLEDGMENTS

The authors acknowledge additional members of the Cornell DARPA Urban Challenge Team: Filip Chelarescu, Daniel Pollack, Dr. Mark Psiaki, Max Rietmann, Dr. Bart Selman, Adam Shapiro, Philipp Unterbrunner, and Jason Wong. This work is supported by the DARPA Urban Challenge program (contract no. HR0011-06-C-0147), with Dr. Norman Whitaker as Program Manager.

REFERENCES

Artes, F., & Nastro, L. (2005). Applanix POS LV 200: Generating continuous position accuracy in the absence of GPS reception (Tech. Rep.). Richmond Hill, ON, Canada: Applanix.

Arulampalam, M., Maskell, S., Gordon, N., & Clapp, T. (2002). A tutorial on particle filters for online nonlinear/non-gaussian Bayesian tracking. IEEE Transactions on Signal Processing, 50(2), 174–188.

Bar-Shalom, Y., Rong Li, X., & Kirubarajan, T. (2001). Estimation with applications to tracking and navigation: Theory, algorithms and software. New York: Wiley.

Bierman, G. (1977). Factorization methods for discrete sequential estimation. New York: Academic Press.

Centa, G. (1997). Motor vehicle dynamics: Modeling and simulation. Singapore: World Scientific.

Cormen, T., Leiserson, C., Rivest, R., & Stein, C. (2003). Introduction to algorithms, 2nd ed. Cambridge, MA: The MIT Press.

Davis, J., Herring, T., Shapiro, I., Rogers, A., & Elgered, G. (1985). Geodesy by radio interferometry: Effects of atmospheric modeling errors on estimates of baseline length. Radio Science, 20(6), 1593–1607.

Felzenszwalb, P., & Huttenlocher, D. (2004). Efficient graph-based image segmentation. International Journal of Computer Vision, 59(2), 167–181.

Ferguson, D., Stentz, A., & Thrun, S. (2004). Pao* for planning with hidden state. In Proceedings of the 2004 IEEE International Conference on Robotics and Automation, New Orleans, LA (volume 3, pp. 2840–2847).

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

Julier, S., & Uhlmann, J. (1997). A new extension of the Kalman filter to nonlinear systems. In Proceedings of the SPIE: Signal Processing, Sensor Fusion, and Target Recognition VI (volume 3068, pp. 182–193). SPIE.

Lemoine, F., Kenyon, S., Factor, J., Trimmer, R., Pavlis, N., Chinn, D., Cox, C., Klosko, S., Luthcke, S., Torrence, M., Wang, Y., Williamson, R., Pavlis, E., Rapp, R., & Olson, T. (1998). The development of the joint NASA GSFC and NIMA geopotential model EGM96 (Tech. Rep. NASA/TP-1998-206861). NASA Goddard Space Flight Center, Greenbelt, MD.

Lundberg, J., & Schutz, B. (1988). Recursion formulas of Legendre functions for use with nonsingular geopotential models. Journal of Guidance, Control, and Dynamics, 11(1), 31–38.

Martin, M., & Moravec, H. (1996). Robot evidence grids (Tech. Rep. CMU-RI-TR-96-06). Pittsburgh, PA: Robotics Institute, Carnegie Mellon University.

Miller, I., & Campbell, M. (2006). A mixture-model based algorithm for real-time terrain estimation. Journal of Field Robotics, 23(9), 755–775.

Miller, I., & Campbell, M. (2007). Rao-Blackwellized particle filtering for mapping dynamic environments. In Proceedings of the 2007 IEEE International Conference on Robotics and Automation, Rome (pp. 3862–3869).

Miller, I., & Campbell, M. (2008). Particle filtering for map-aided localization in sparse GPS environments. In Proceedings of the 2008 IEEE International Conference on Robotics and Automation, Pasadena, CA (pp. 1834–1841).

Miller, I., Lupashin, S., Zych, N., Moran, P., Schimpf, B., Nathan, A., & Garcia, E. (2006). Cornell University's 2005 DARPA Grand Challenge entry. Journal of Field Robotics, 23(8), 625–652.

Psiaki, M., & Mohiudden, S. (2007). Modeling, analysis and simulation of GPS carrier phase for spacecraft relative navigation. Journal of Guidance, Control, and Dynamics, 30(6), 1628–1639.

Russell, S., & Norvig, P. (2003). Artificial intelligence: A modern approach, 2nd ed. Upper Saddle River, NJ: Prentice Hall, Pearson Education, Inc.

Saastamoinen, J. (1972). Atmospheric correction for the troposphere and stratosphere in radio ranging of satellites. In S. Henriksen, A. Mancini, & B. Chovitz (Eds.), Geophysical Monograph Series, 15, 247–251.

Savage, P. (1998a). Strapdown inertial navigation integration algorithm design, Part 1: Attitude algorithms. Journal of Guidance, Control, and Dynamics, 21(1), 19–28.

Savage, P. (1998b). Strapdown inertial navigation integration algorithm design, Part 2: Velocity and position algorithms. Journal of Guidance, Control, and Dynamics, 21(2), 208–221.

Sleewaegen, J.-M., De Wilde, W., & Hollreiser, M. (2004). Galileo Alt-BOC receiver. In Proceedings of ION GNSS 2004, Long Beach, CA.

Sukthankar, R. (1997). Situational awareness for tactical driving. Ph.D. thesis, Robotics Institute, Carnegie Mellon University, Pittsburgh, PA.

Trimble (2004). AgGPS 252 Receiver User Guide Version 1.00, Revision A, Sunnyvale, CA.

Willemsen, P., Kearney, J. K., & Wang, H. (2003). Ribbon networks for modeling navigable paths of autonomous agents in virtual urban environments. In Proceedings of IEEE Virtual Reality 2003 (pp. 22–26). IEEE.

Wong, J. (2001). Theory of ground vehicles, 3rd ed. New York: Wiley.