15

MITRE Meteor: An Off-Road Autonomous Vehicle for DARPA's Grand Challenge

Robert Grabowski, Richard Weatherly, Robert Bolling, David Seidel, Michael Shadid, and Ann Jones

The MITRE Corporation, 7525 Colshire Drive, McLean VA, 22102

Summary. The MITRE Meteor team fielded an autonomous vehicle that competed in DARPA's 2005 Grand Challenge race. This paper describes the team's approach to building its robotic vehicle, the vehicle and components that let the vehicle see and act, and the computer software that made the vehicle autonomous. It presents how the team prepared for the race and how their vehicle performed.

15.1 Introduction

In 2004, the Defense Advanced Research Projects Agency (DARPA) challenged developers of autonomous ground vehicles to build machines that could complete a 132-mile off-road course. 195 teams which applied – only 23 qualified to compete. Qualification included demonstrations to DARPA and a ten-day National Qualifying Event (NQE) in California. The race took place, on October 8 and 9, 2005 in the Mojave Desert, over a course containing gravel roads, dirt paths, switchbacks, open desert, dry lakebeds, mountain passes, and tunnels.

The MITRE Corporation decided to compete in the Grand Challenge in September 2004 by sponsoring the Meteor team. They believed that MITRE's work programs and military sponsors would benefit from an understanding of the technologies that contribute to the DARPA Grand Challenge.

This paper describes the MITRE Meteor team's approach to building its robotic vehicle, the resulting vehicle and associated sensors and systems, and the results of their efforts in the Grand Challenge race.

15.2 Approach

The Meteor team first decided on some underlying approaches to building the robot:

- Create a robot that could act as a test vehicle for missions, such as convoy leader/ following (Cheok, Smid, Kobayashi, Overholt & Lescoe, 1997), surveillance (Saptharishi, Bhat, Diehl, Dolan & Khosla, 2000), unmanned transport (Sotelo, Rodriguez & Magdalen, 2000), and cooperative robot missions (Sato et al., 2004).

M. Buehler, K. Iagnemma, and S. Singh (Eds.): DARPA'05, STAR 36, pp. 483–516, 2007.
springerlink.com

Fig. 15.1. The MITRE Meteor starting the finals of the 2005 DARPA Grand Challenge

- Create a robot that represents an affordable solution for sponsors, such as the Department of Defense and Department of Homeland Security.
- Remember that the robot must be built and tested in less than ten months.
- Focus on the most challenging aspect of the race – sensing and maneuvering.
- Employ COTS solutions wherever possible don't create where commercial solutions suffice.
- Use an incremental model-simulate-test approach. Build a model suitable to the current task. Verify and tune the model using simulation and replay. Test the model and system in real situations, and then use the results of the testing to adjust the model as necessary.
- Actively manage risk. Fundamentally, the challenge is straightforward and does not require a complex solution. Rather, the contest exposes the need to manage interdependencies among multiple systems.

15.3 Vehicle

The first step in producing a rugged autonomous vehicle is platform selection. Several vehicle types were considered: Racing buggy, all-terrain vehicle, four-wheel drive (4WD) sports-utility vehicle, and 4WD pickup. Investigation of the 2004 Grand Challenge results pointed to a platform capable of off-road travel, able to support multiple sensors and processors, and able to withstand desert temperatures.

The vehicle should also be easy to transport and operate. To reduce the cost of deployment, it should be drivable by a human operator. To improve safety during testing, a human operator should be able to override computer controls.

A commercially available vehicle that could be retrofitted with drive-by-wire capabilities was preferred – this would let the team focus more quickly on relevant issues. Similarly, a solution that could be adapted to any vehicle with drive-by-wire capability (such as, Humvees or tracked vehicles) was preferred.

The team chose a 2004 Ford Explorer Sport Trac pickup from a local dealer (Figure 15.1). It has reasonable off-road capability and a sufficiently cooled interior for computing equipment. Ford was selected because it is most understood by the selected drive-by-wire vendor.

15.3.1 Vehicle Modifications

The Grand Challenge required three major modifications to the vehicle: A drive-by-wire capability, an expanded electrical power system, and a chassis lift.

The Electronic Mobility Controls Corporation (EMC) designed and installed the drive-by-wire capability. They modified the transmission and steering column (Figure 15.2). A servo, mounted inline with the steering wheel, accomplishes turning. A second servo, attached to the firewall, controls the throttle and brakes. The installation also included an electrical override console with controls for the steering, throttle, and brake. Using this, a human safety operator can take control of the vehicle with the touch of a button. The capability also provides a safety override so that an operator in the driver's seat can always operate the brakes manually.

The second modification was installing a heavyduty 220-A alternator and power bus. This eliminated the need for additional batteries or an external generator, and easily handled the load of the sensors and processors. Power passes from the alternator to the rear processing rack via a high-current dc bus. A 3000 W dc-to-ac converter provides 120 V for computers and sensors.

The last major modification was the installation of a lift kit on the vehicle. An analysis of the 2004 route raised concerns about whether the vehicle could clear moderate-sized rocks and debris (Urmson et al., 2004; Ozguner, Redmill &

Fig. 15.2. EMC provided the drive-by-wire capability using a steering servo, a throttle/brake servo, a control console, two servo computers, a battery backup, and a manual override

Broggi, 2004). So, a 4 in. suspension lift was donated by SuperLift, an off-road retrofitter. Additionally, four 30 in., offroad Super Swamper tires were installed to increase traction in rugged terrain.

By employing these COTS components, the team was able to start testing on a reliable platform within two months.

15.3.2 Sensor and Processor Suite

Similarly, the team selected COTS sensors and computers. The sensing centers around eight laser sensors were oriented in different directions and mounted on the top and front of the vehicle (Figure 15.3). Laser range finders provide a two-dimensional range/distance map out to 40 m with a 100° field of view. These lasers are the staple of the robotics community and were used by almost every entrant in the Grand Challenge. Meteor utilized two models of the laser sensor from SICK; one operating at 5 scans per second, the other at 75 scans per second.

The laser range finders were divided into classes based on their orientation. Two vertically mounted lasers provide information about the ground plane. Three lasers, mounted horizontally at different angles, provide both short- and long-range detection of road obstacles. A sixth horizontal laser is mounted on gimbals, and points dynamically to compensate for vehicle pitch and terrain variations. Finally, two downward-looking lasers are mounted on the roof to detect road characteristics.

Two sensor racks were built to mount the sensors to the outside of the vehicle. The largest sensor rack is mounted on top of the vehicle. It provides the platform for mounting the global positioning system (GPS) receivers, inertial navigation system (INS), and magnetic compass. This positioning allows the greatest visibility to satellites and moves the sensors away from vehicle noise. Additionally, the two down-looking lasers are mounted at the front of the rack to give them the greatest clearance and visibility of small ground obstacles near the vehicle. The rack is held to the roof struts using four U-bolts, so that the entire assembly can be removed and stored in the vehicle cargo bay during shipping.

A second sensor rack utilizes a grill extension. Shelves added to the front-grill guard house the horizontal and vertical obstacle and road scanners. The rack is mounted rigidly a few inches in front of the grill to allow engine air flow. The headlight guards were removed to prevent misalignment in the event of collision. The headlights were expendable and the sensor system should prevent a head-on collision.

The processing infrastructure is mounted in the cabin of the vehicle. The rear passenger seat was partially removed, and a rack assembly was built to hold the processing and display components. The entire rack is shockmounted in five places. The computing infrastructure is a multiboard militaryhardened VME computer array of 1.8 GHz Pentium processor boards, connected by a gigabit Ethernet network, and mounted in a 19 in. rack. Only four of the available nine slots were needed for the competition. One additional card was dedicated solely to logging and display during testing. The rear passenger area also contains a

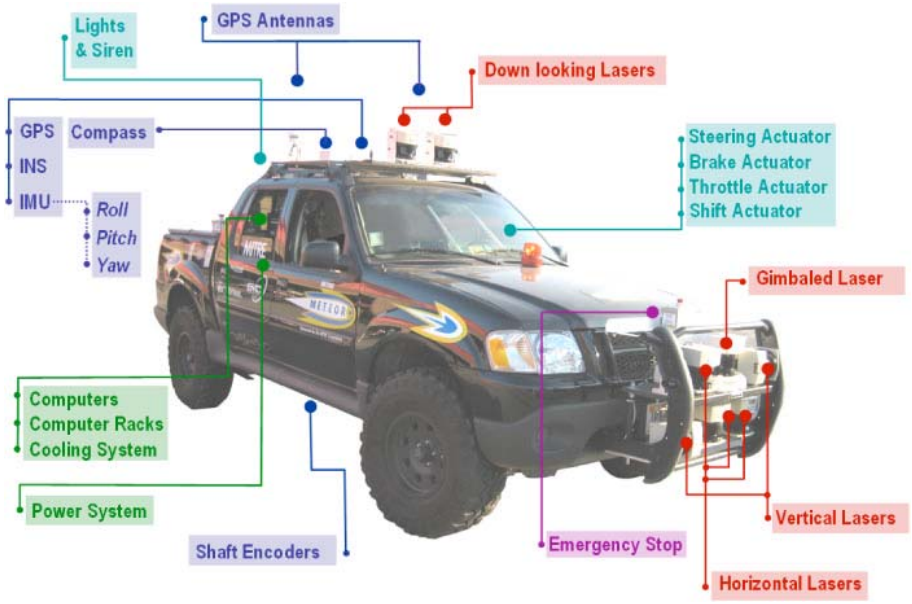

Fig. 15.3. MITRE Meteor layout. Meteor has an array of COTS components. Positioning is by three GPS units, an INS unit and a compass on the roof. Four horizontal lasers detect road obstacles. Two vertically-mounted lasers provide terrain information. Two down-looking lasers detect small objects and negative spaces. A computer assembly is in the back seat.

rack-mounted monitor, keyboard, and mouse, to permit an observer to oversee and interact with the system during testing.

15.4 Software Architecture

Experience shows that software evolves continually until it is retired. Characterizing axes of change reveals what can and cannot be specified by software architecture. For this effort, two axes of change were clear: (1) The controller would have a number of quasi-independent activities, whose algorithms could change during the development process, and (2) the population and relationship between these activities would evolve. Therefore, an agent-based architecture (Minsky, 1985) was chosen.

Experience also shows that building the software scaffolding (test harnesses, stubs for yet-to-be-built components, verifiers, etc.) surrounding a software product can be time consuming. To reduce this overhead cost, two themes were applied to the design: Location transparency and employment transparency. That is, use the same piece of code in as many different places as makes sense, and for as many different purposes as is reasonable.

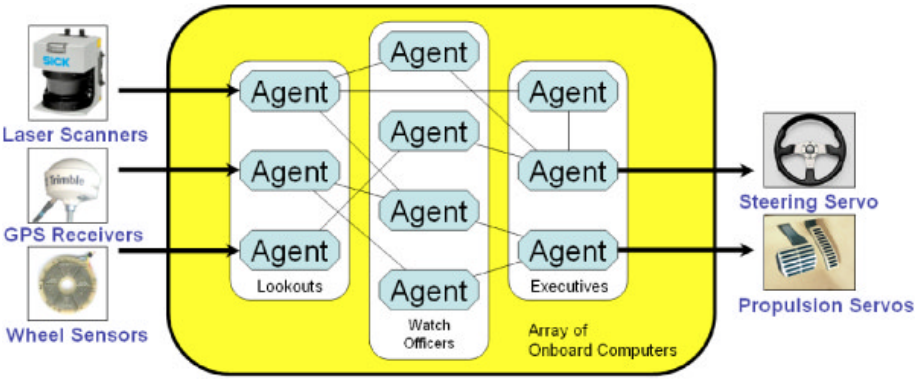

Fig. 15.4. Basic agent makeup. Lookout agents manage sensors, and convert raw sensor information into system messages. Watch Officers process and fuse sensor data from Lookouts into higher-order information. Executives use Watch Officer products to make decisions that ultimately control the actions of the Meteor.

15.4.1 Agent-Based Approach

The entire software architectures consists of agents that receive state information, add value to that input, and pass it to the remainder of the system (Figure 15.4). Agents are of three basic types: "Lookouts," "Watch Officers," and "Executives." Lookout agents manage sensors and convert raw sensor information into system messages. Watch Officers process and fuse sensor data from one or more Lookouts to provide higher-level information, such as vehicle pose, obstacle definitions, and ground plane estimates. Executives use Watch Officer products to make decisions that ultimately control the actions of Meteor.

Specialized "executives" in the program map the vehicle state to motion (Figure 15.5). A "Captain" takes the provided Route Definition Data File (RDDF) and generates a lane in which the vehicle can operate. A "Navigator" takes the lane and obstacles and generates a viable path. It then uses the path to determine a desired speed and direction based on the vehicles current state. Desired speed and direction are passed to a "Helmsman" which converts these parameters into commands that adjust the two voltages that drive the EMC system. An "Admiral" agent gives a final go/no-go signal based on the state of the emergency stop system. While components within this architecture evolved, the basic architectural design remained unchanged.

15.4.2 Location Transparency

Vehicle control is achieved by agents that communicate with each other using UDP messages. Raw sensor input and aggregate state are manipulated as Java classes, serialized for transmission as messages, and then shared between computers via a fast Ethernet switch. The agent state is mirrored on each computer and kept current through a constant assertion methodology. The vehicle state,

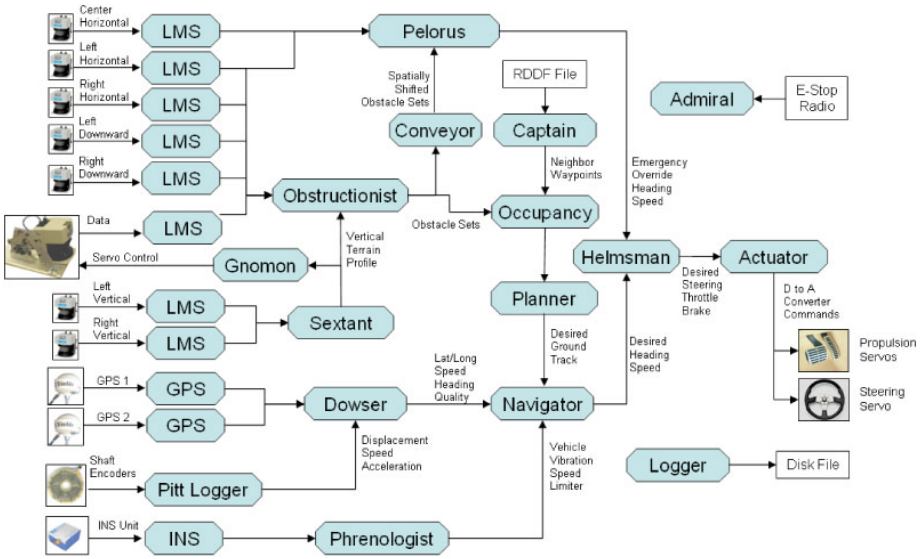

Fig. 15.5. Software architecture implementation. An agent-based architecture converts raw sensor information into voltages to drive the robot. "Lookouts" convert raw sensor data into system messages. "Watch Officers" turn raw information into information, such as, obstacles and position. The "Captain" takes the RDDF and defines the next achievable point and the lane in which the vehicle can operate. The "Planner" uses the lane and obstacles generated by the lasers to generate a plan. The "Navigator" determines desired speed and direction based on the plan and the vehicle's state. The "Helmsman" converts the speed and direction into commands that cause the EMC system to produce motion. The "Admiral" monitors the E-Stop radio and gives the master-go signal.

sensor values, and obstacle inventory are calculated and reported periodically. The advantage of employing a message transfer protocol is that agents can be located anywhere without modifying the agent code. This permits load balancing among multiple computers. Initially, all of the components required to drive the vehicle ran on a single laptop computer. As functionality was added, agents were distributed among several computer boards. High demand agents, such as the occupancy map Watch Officer, were allocated their own computers. The motion model and obstacle filters require less computation and share the same computer. One of the biggest processor loads is the graphical display software used by the operator to monitor progress during development — it was allocated its own computer.

15.4.3 Employment Transparency

A unique feature of this architecture is that it is insensitive to the source of input. Because the system state is continuously shared, the system behaves nearly

identically whether driven by simulation, replay, or live data. During testing, all sensor data and interagent messages are recorded. Posttest analysis examines performance in replay, and observes system behavior. By filtering selected messages, it is also possible to exploit real-world data to evaluate modifications to existing algorithms while in the lab. For example, by filtering out path messages, a new planner agent can generate live plans based on the information generated from a previous test.

The system can also be driven completely from simulated sources (Figure 15.6). Simulated sensors produce raw information from external sources, such as a ground truth map. These signals are passed through the system producing the same control voltage messages. These control voltage messages are passed to a simulated vehicle model, that in turn generates messages that feed back into the system, such as vehicle location, steering angle, and encoder values.

Because the operational software functions in the same way in simulation, replay, or actual use, each of these modes can support the other. As new agents were developed, they were first run in the simulator. When behavior was acceptable, they were tested in the field. Data from live runs were used to tune the agents and simulation models.

Location and employment transparency reduced testing time by recording all aspects of a run, evaluating the results, and testing resultant changes in simulation. Even though only 10 days of desert testing was conducted, the design permitted significant improvements to the system.

15.4.4 Development Environment

The software was developed for the Fedora Core 3 Linux operating system using the Java 5 programming language. The development environment employed Sun

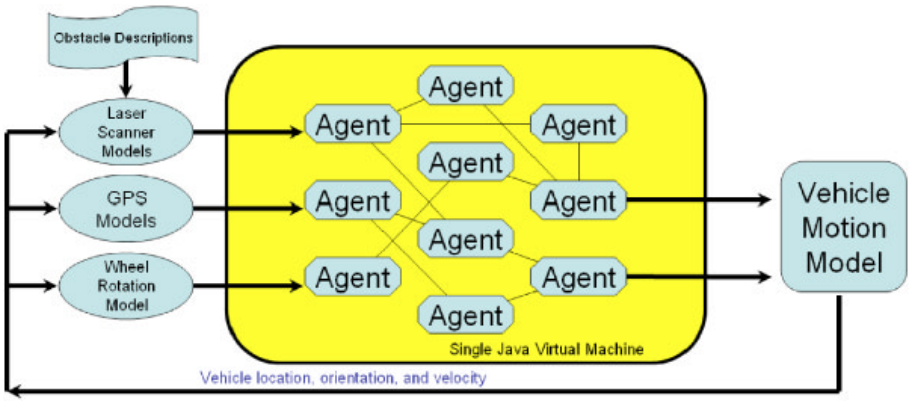

Fig. 15.6. Employment transparency. Employment transparency means that the system is unaware of whether it is being driven by real, simulated, or replay data. Here, a simulated vehicle model adds a feedback mechanism to allow simulation of events.

JDK 1.5, Subversion, Ant, and Eclipse. Each of these technologies, is proven, readily available, and carries a low maintenance and registration overhead.

15.5 Maneuvering and Navigation

The most rudimentary task facing a vehicle in the Grand Challenge is the ability to follow a series of waypoints. DARPA defined the route for the race as a series of about 3,000 connected latitude/longitude positions that wind through the desert. DARPA specified that the points would define a path with no intervening impassable features. They also defined the maximum lane width and maximum safe speed for each segment. Each vehicle then needed to compute its own position, the position of the next point, and a strategy for getting there.

15.5.1 GPS Positioning

Positioning for Meteor is accomplished by three GPS units. The primary GPS receivers are two Trimble AgGPS132 differential GPS units with differential corrections from the Omnistar subscription service. With proper sky visibility, the Omnistar service improves position accuracy to within 5-10 centimeters at a rate of ten times per second.

Each Trimble unit reports signal strength and signal-to-noise ratio. These measurements determine whether a GPS unit is reporting accurate readings. Testing showed that the units were not consistent with one another; so to accept positions as valid, both measurements were required to be above a chosen threshold. In most cases, the measurements dropped quickly enough to indicate a loss of valid position before the system has deviated too far.

Figure 15.7(a) shows the GPS signal-to-noise and signal strength measurements for a run during the NQE. Here, the GPS dropped below threshold many times along the track, with the tunnel being the longest. Circles correspond to the five selected points (pluses) in the data with the second being immediately before the tunnel. The chosen threshold for the signal-to-noise ratio was 9 and the signal strength was 110. During the NQE, both values were raised slightly.

The GPS units seemed reliable during testing up to the NQE, including testing in the deserts of Yuma and Primm. However, after anomalies during the first NQE run, their output was re-examined. During practice testing and replay of the first run, the vehicle position jumped a few meters and returned even when the vehicle was stopped. Between Runs 1 and 2, static GPS tests were conducted. Figure 15.7(b) (top) shows the results of taking 2,500 GPS samples with the vehicle stationary. Instead of positions randomly centered about the origin of the robot, several correlated clusters formed a pattern about the origin (dotted circles). Moreover, the plotted histories of the pattern (Figure 15.7(b) bottom) showed that, instead of being randomly distributed, they dwelled at one point before jumping to the other. When running, the vehicle had enough time between jumps to try to compensate for the new errors. The vehicle appeared to jog in and out of the lane as it was driving.

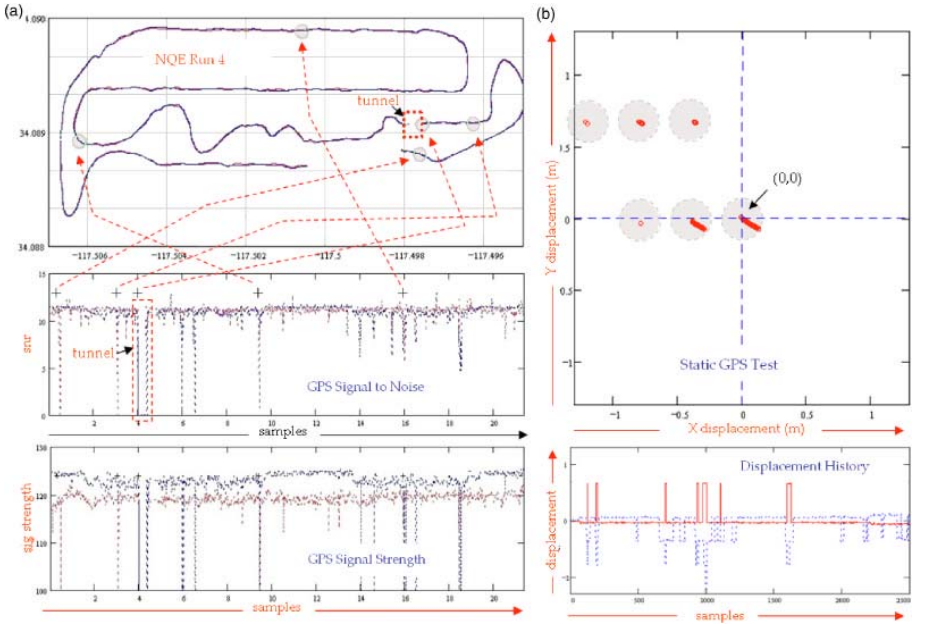

Fig. 15.7. GPS verification. GPS reliability is determined by assessing signal strength and signal-to-noise ratio reported by GPS units. If the ratios drop below a set threshold, their reported positions are not trusted, and Meteor is guided by a dead reckoning model. (a) Signal-to-noise ratio and signal strength plots during a run at the NQE. Circles correspond to selected points on the plots. Note the complete drop out inside the tunnel. (b) 2,500 GPS readings taken while the robot was stopped during testing at the NQE. Note the apparent spatial periodicity implying multipath. Below is the same series as a function of time and displacement.

The regular pattern of clusters suggested a strong influence from multipath, perhaps from metal structures, such as, fences and bleachers that surrounded the NQE. The other interesting phenomenon was the correlation between readings within the clusters. Even with multipath, the readings should have been random about a point. Instead, they seemed smeared along a line oriented in a south/south east direction. No explanation was discovered for this behavior.

A third GPS is a MIDG-2 INS. This GPS unit is augmented by an internal inertial measurement unit that maintains some location ability during GPS outages. If both Trimble GPS units are above their threshold, the MIDG GPS unit serves as a tiebreaker. Tiebreaking prevents arbitrarily jumping back and forth between units. Even though both are accurate, they differ slightly and jumping back and forth introduces high-frequency position noise.

Heading is determined by comparing successive positions. This method proved to be very accurate at speeds above 3 miles per hour, but ineffective at lower speeds. Initially, a Honeywell magnetic compass provided an alternative source of

heading. However, it was abandoned because it required calibration in different locations (Nevada and California), had high latency (degrees per minute), and drifted as much as 10° with the vehicle stopped.

15.5.2 Motion Model

Even with differential GPS in wide-open spaces, GPS experienced periodic and sometimes sustained losses. To overcome these losses, a simple bicycle model was used to predict the vehicles' location.

Vehicle displacement is measured via a multielement quadrature shaft encoder, mounted directly to the vehicle drive shaft. We assume no differential slip or tire slip. A similar encoder is mounted to the steering column, and provides steering wheel angle. This is input to a lookup table that maps steering column angle to turn radius. The table was populated experimentally. The turn radius and displacement measurements are passed to the bicycle model, which produces a new position estimate. Since the bicycle model is described in many robotic papers, we omit it here.

If either Trimble GPS unit produces a valid reading (above the detection threshold), the deadreckoning position is not needed, and its current estimate is set to the measured GPS position. If no valid GPS reading is available, the dead-reckoning model becomes the position provider.

Testing showed that, for most of the surfaces, the dead-reckoning model could run for about 100 meters without a GPS lock and keep the vehicle within the boundaries of a road. This distance drops quickly if the vehicle must perform many turns. While not sufficient to track position for a long distance, the model was sufficient for the types of GPS losses expected in the course.

15.5.3 Steering Gain

Once the vehicle is able to determine its own position, it has a context to follow a route. A popular approach for route following is to continuously point the vehicle toward the next waypoint in the route definition. This approach has subtle but significant drawbacks. The error between the actual steering angle (the way the vehicle is pointing) and the derived steering angle (the angle to the waypoint) increases as the vehicle nears the waypoint. This effect, termed as steering gain by the team, can lead to oscillations when the vehicle is close to a waypoint. The phenomenon is exacerbated by small perturbations in reported vehicle position. Instability increases quickly with speed and can lead to unsettling performance above 20 mph. Steering instabilities can lead to other effects, such as excessive roll, which can lead to errors in obstacle detection.

To address this, a "carrot" mechanism was introduced to regulate the steering gain of the robot (Figure 15.8a). The carrot acts as a virtual waypoint that moves back and forth along the intended path of the robot. Instead of steering toward the next waypoint, the vehicle steers toward the carrot. Since steering gain is a function of vehicle speed, carrot distance is a function of vehicle speed. As vehicle speed increases, the carrot moves farther from the vehicle. As the vehicle slows, the carrot moves closer.

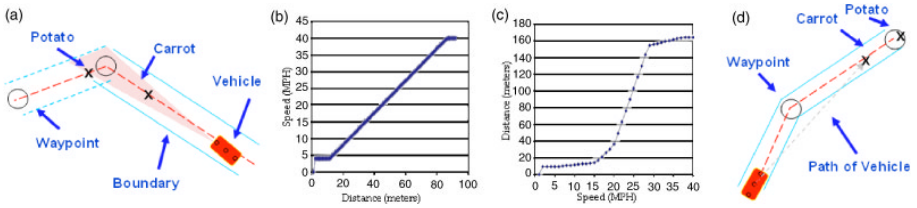

Fig. 15.8. Carrot maneuvering. Steering stability is achieved using a carrot and potato. (a) The carrot maintains its distance in front of the vehicle based on vehicle speed. Maximum vehicle speed is a function of path visibility and is determined by the potato. The potato defines visibility, the carrot follows the path, and the vehicle follows the carrot. (b) A curve mapping the distance to the potato to maximum vehicle speed. (c) A curve mapping actual vehicle speed to carrot distance. (d) An example where a large carrot distance can cause a slight shaving of the defined route.

The carrot mechanism provides stable steering across the range of vehicle speeds but does not determine target speed. This is accomplished by the "potato" which, like the carrot, is a point that moves along the intended path of the robot. The potato stays ahead of the vehicle on the path at a point that represents the limit of vehicle path visibility. The term path visibility captures the notion of how far the vehicle can see down a 2-dimensional pipe defined by the intended path. Notwithstanding the effect of obstacles, the vehicle sets its speed based on the distance to the potato (Figure 15.8b). If the path is long and straight, the potato is far away and the vehicle can travel swiftly. If the path is windy or contains a sharp curve, the potato is close and the vehicle slows.

Path visibility distance is determined by summing the distance along the path starting from the vehicle, while simultaneously summing the absolute angle between those segments. When the absolute sum of the angles exceeds a threshold, path distance summation stops. The summation method accounts for sharp curves, as well as the cumulative effect of shallow curves.

The functions that map vehicle speed to carrot distance (Figure 15.8c) and potato distance to target vehicle speed are constructed of linear segments. These mappings were determined first in the system simulator and then experimentally on a dry lakebed in Nevada. For a given speed, the carrot distance was manually adjusted out until the vehicle steering stability (degree of overshoot under a perturbation) was satisfactory. Next, the potato speed mapping was adjusted to assure that the vehicle slowed properly when approaching a turn. Aside from traction problems, entering a turn too fast forces the carrot around the corner too soon and causes the vehicle to shave the inside of the turn (Figure 15.8d).

The biggest advantage of using path visibility is that the vehicle naturally slows as it approaches large turns, and then speeds up as it passes through the turn. Note that this path is generated by the planner (discussed later), and not the path given by the route definition (RDDF file).

15.5.4 Additional Speed Limiting

In addition to path geometry, several additional mechanisms regulate vehicle speed. While speed is important for competing in the Grand Challenge, it increases risks inherent in a large moving vehicle (Gillula, 2005). Simply increasing the speed without addressing safety, stability, and sensor range fails to recognize the dangers inherent in large robots. Higher speed reduces the distance available to react to an obstacle, decreases sensor fidelity as samples are taken over a larger area, and consequently decreases confidence in a selected action. At higher speeds, vehicles are more likely to tip over or swerve off the road from an unexpected steering correction. In the event of a collision, higher speed increases the momentum of a vehicle; increasing the likelihood of damage. This is evident from the damage to the parked cars at the NQE from robots colliding with them at low speeds. DARPA defined the maximum safe speed for each segment of the route. The team chose to respect this value.

As discussed above, path visibility is the primary factor for setting the speed along a segment. It determines speed based on how far it can "see" along the geometry of the path and slows the vehicle before entering sharp curves or a complex calculated path.

Another speed consideration is congestion where the vehicle slows when it perceives that it is entering an area with high density of objects, even if it has determined a clear path through it. The robot should not drive through an obstacle field at high speeds, but should anticipate danger and react appropriately. Congestion is calculated by summing the number of occupied cells (discussed in Section 15.7.2) about a narrow rectangular space along the path in front of the vehicle. If the path is congested, the vehicle slows.

A similar safety mechanism slows the vehicle if it senses excessive vibration. Not only can excessive vibration damage sensitive components, such as computer disk drives and connectors, but it can also increase stopping distance and add steering instabilities. Moreover, vibration can be a precursor to other dangers. Testing in the desert showed that seemingly small desert shrubs could accumulate large amounts of dirt at their base. Meteor hit one at 25 mph and the resultant jolt broke off one of the shock mounts and forced the vehicle several meters off the road. Thus, vehicle speed is reduced based on frequency and magnitude measurements from the INS vertical accelerometers. Like many such mappings, a transfer function was defined and tuned based on driving over a range of different road conditions in the desert.

The last speed control mechanism is engaged by the reactive system. Normally, the vehicle follows a path generated by the planner, and speed is based on path visibility and congestion. However, a reactive system engages when the planner misses an obstacle and it endangers the hull. The reactive systems steers the vehicle away from the obstacle along the nearest open path. At the same time, it slows the vehicle to provide a better opportunity to maneuver. When the reactive system engages, it slows the vehicle to 4 mph until it has cleared the obstacle. A small dead band and hysteresis prevents premature triggering and release.

These methods of reducing speed in anticipation of danger cause the robot to err on the side of caution. Because of limited development time and access to desert terrain, speed control mechanisms were only roughly tuned. As a result, the vehicle was able to achieve its maximum speed only few times during the NQE and race. However, these speed mechanisms were the likely reason that the vehicle suffered no major incidents or collisions during testing and competition.

15.6 Obstacle Processing

The primary function of the laser rangefinders is to detect obstacles that lie directly in the path of the vehicle. Initially three fixed lasers were mounted on the front at slightly different angles to determine obstacles immediately in the path of the robot (Figure 15.9a). One laser pointed directly out while the other two were oriented about 3 degrees above and below the horizon. Different angles were selected to maintain acceptable sensing distances even as the vehicle was driving over and down elevated terrain. Early tests showed that the small pitches in terrain could cause the laser to miss obstacles as large as trashcans as it passed over the object or struck the ground just before it. Just before deployment, a steerable laser was developed that could scan up and down, reducing the need for fixed lasers. However, time constraints prevented comparison between the two approaches and both were used.

15.6.1 Obstacle Generation

Each laser produces a series of 400 range readings with an angle increment of 0.25 degrees. With four lasers reporting at 5 times a second and the remaining

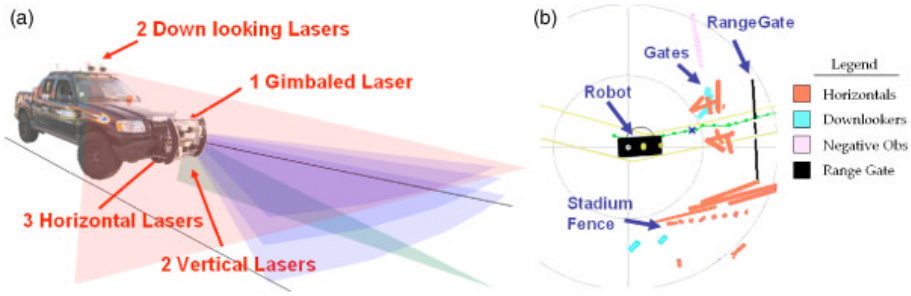

Fig. 15.9. Sensor types. (a) Meteor carries four classes of laser sensors. Three fixed horizontal lasers mounted at slightly different angles detect obstacles directly in front of the vehicle. A gimbaled horizontal laser tracks the horizon providing the greatest sensing distance. Two vertically-mounted lasers detect road terrain and provide dynamic range gating to the horizontal lasers. Two down-looking lasers detect small ground obstacles and negative terrain. (b) Line objects generated by each laser are displayed during testing. This is a view of Meteor as it is about to pass through the first gate at the NQE.

four reporting at 75 times a second, raw message traffic is large. To reduce message traffic, each laser scan is converted to a series of line segments before being passed through the system. Conversion from raw laser data to line segments provides a more compact representation - generally less than 10 segments per scan. This representation also reduces the processing complexity of downstream agents analyzing these segments. In fact, it increased the performance of occupancy map generation by an order of magnitude. Reducing the fidelity of laser scans also reduces the detail and look of the generated map, but it had little effect on the ability to represent traversable terrain.

Objects detected by lasers fit into three categories – point obstacles, line obstacles, and vistas. Objects are generated during each pass of a laser scan by clustering continuous readings together using a simple state machine. Contiguous readings that are less than the maximum range are clustered into obstacles and represent solid obstacles. Contiguous readings that are all at maximum range are clustered as vistas and represent open space.

A post-scan filter was also added to remove some objects based on their projected size. That is, small line and point obstacles close to the vehicle indicate objects that are only a few centimeters in width and are more likely to be noise or debris like falling leaves.

Figure 15.9b shows the line obstacles and vistas produced by each of the lasers as Meteor passes between two gates at the beginning of the NQE. Each is coded based on their source. This instantaneous view suggests how the vehicle is responding to the most current event and is an essential tool for understanding and development.

15.6.2 Vertical Lasers and Dynamic Range Gating

A complication of mounting horizontal sensors low in front of the vehicle is that the sensor scan might hit the ground. A horizontal laser pointing downward will probably return non-maximal readings across the entire scan making it difficult to distinguish between an obstacle and the ground. If uncorrected, the robot would always try to avoid the ground in front of it. A naive model would assume that the world was flat and horizontal then calculate where the laser would intersect the ground and reject all readings greater than that value. However, this assumption limits the terrain where the vehicle could operate. Small crests or depressions would produce unwanted results. Moreover, passing over rough terrain could introduce large perturbations in pitch and roll which have a similar effect to changing terrain for the laser scanners.

To compensate for the effects of non-flat terrain, two lasers were mounted vertically to the front grill (Figure 15.10d). Each vertical laser produces a scan that starts from directly beneath the vehicle and scans outwards along its major axis (Figure 15.10e left). In the absence of obstacles, the scan represents the contour of the ground directly in front of the vehicle (Figure 15.10c).

The vertical scan is used by the horizontal lasers to generate a range gate – a single value that represents the distance from that sensor to the ground plane. Horizontal range values beyond the range gate are likely to be caused by intersection

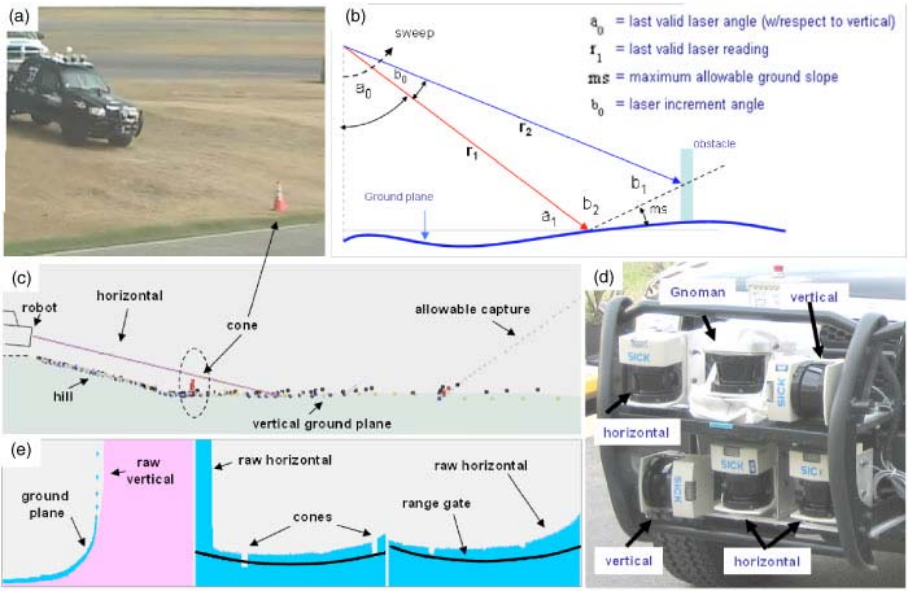

Fig. 15.10. Vertical lasers. (a) Meteor travels down a steep hill at the NQE. (b) A slope filter removes obstacle readings from the vertical scan based on the slope change between readings to generate a ground plane estimate for range gating. (c) A projection of the filtered ground plane just before Meteor begins to drive down the hill. Note that the cone at the bottom of the hill (dotted ellipse) has been removed from the scan. (d) The physical mounting of the lasers. A vertical laser collinear to the horizontal lasers it services. This alignment makes the range gate a simple geometric lookup in the vertical scan. (e) Raw vertical laser scan (left pane, shaded) and projected ground plane (bowed line). Raw range scans from two horizontal lasers (center and left pane shaded). The crossing lines are the dynamic range gate. Note that the cones on either side of the bottom of the hill are visible as gaps in the raw scans, and are starting to show up as obstacles in two of the horizontals.

with the ground and are rejected (Figure 15.10e right). Those shorter than the range gate are likely to be the result of a true obstacle. Since both the vertical and horizontal lasers are mounted rigidly and the vertical lasers are mounted beside the corresponding horizontal lasers, the dynamic range gate is insensitive to whether the changing profile is caused by terrain changes or vehicle transients. Moreover, by mounting them collinear to one another, range gate calculations are reduced to a simple index lookup and make fewer assumptions about the geometry of the vehicle.

To provide an accurate range gate, the vertical laser must filter out any obstacles that lie directly along the axis of the vehicle. Without this filter, the obstacles would be incorporated into the ground plane estimate, and could result in an erroneous range gate that blinded the horizontal scans of true obstacles.

Obstacles are rejected by looking at the rate at which the ground plane height (a function of range) changes over the scan (Figure 15.10b). If the height changes by a large amount, it is probably an obstacle. If the rate change is small, it is likely the road contour. Based on experiments in the desert, a rate change was selected that corresponds to a road slope of 15 degrees (the largest test measurement of road slope was 9 degrees).

To reduce computational complexity and latency, this rate change is expressed in terms of the previous range value and current sweep angle. If the scan is started from close to the vehicle, the road profile is always monotonic. This allows filtering to be accomplished in a single pass as the laser scan is read. We calculate the minimum range reading by applying the Law of Sines (Figure 15.10b, Equation 15.1).

$$r_2 = r_1 \cdot \frac{\sin(b_2)}{\sin(b_1)} = r_1 \cdot \frac{\sin(90 - ms + a_0)}{\sin(90 + ms - a_0 - b_0)} \tag{15.1}$$

If there are no obstacles on the road, each successive range reading should exceed this minimum threshold. If the threshold is not met, the ground plane is estimated (Equation 15.2). Since no information about the terrain behind an obstacle is available, the ground shadowed by any obstacles is assumed horizontal.

$$\begin{aligned} &\text{if } (\mathrm{r}_{\text{measured}} > r_2) \left\{ a_0 = a_0 + b_0 \;:\; r_1 = r_2 \;:\; \mathrm{r}_{\text{ground}} = r_2 \right\} \text{ (measured)} \\ &\text{else} \left\{ \mathrm{r}_{\text{ground}} = \frac{h_{laser}}{\cos(a_0 + b_0)} \right\} \text{ (estimated)} \end{aligned} \tag{15.2}$$

Since an obstacle may shield many points, the estimate will continue until a valid ground measurement is obtained. However, since the ground could have continued to rise in this shielded region; the acceptable threshold is slowly increased to allow it to deal with rising terrain beyond the obstacle. However, this too was tempered. For example, at the NQE the first series of gates shadowed a large region beyond the vertical lasers as it passed over them. The next readings that were not shadowed were from the bottom of the bleachers directly behind the gates (another obstacle). These points were low enough that they could be mistaken for a slow rise starting at the gates and cause an erroneous indication of ground plane. To alleviate this situation, the ability to recapture the ground plane beyond large occlusions was limited.

Figure 15.10 shows the vertical lasers in action as the vehicle is about to travel down the steep hill just before entering the hay bales (Figure 15.10a). Figure 15.10c shows the projection of both vertical laser scans reaching below and outward from the vehicle. The dots projecting away from the vehicle show the points accepted as part of the ground plane except those in the dashed circle that were rejected as an obstacle. In this view, the reading was taken at the top of the hill looking down about 5 meters from the bottom at a slight angle to the road path. The vertical scan passed across one of the cones. Because the instantaneous

ranges to these points do not increase fast enough, they were rejected. The solid line projecting out of the robot represents the horizontal lasers and shows where they intersect the ground plane. Figure 15.10e (left) shows the raw scan from one of the vertical lasers (light shade) and the corresponding ground plane estimation (darker line). Figure 15.10e (center and right) shows two of the raw horizontal scans (shaded) with the corresponding range gate drawn as a solid black curve across the scan. Note that the range gate is just above the main part of the scan effectively eliminating all the ground readings but is low enough to indicate the presence of the two cones located at the bottom of the hill on either side of the lane.

15.6.3 Scanning Laser

An articulated laser, called the Gnomon, was developed that could scan up and down using a simple 4-bar mechanism. The articulated laser could be adjusted to point up or down 10 degrees about the horizontal.

To maximize sensing distance, the Gnomon was pointed just above the horizon. The Gnomon agent determines the horizon by scanning the ground plane developed by the vertical lasers to find the point where the range readings reach their maximum. Since the ground plane reading is filtered, this maximal reading represents the farthest point in front of the vehicle. The Gnomon is then steered a few degrees above that number to clear any ground clutter.

Steering to the horizon is not always the best action for a laser. As the robot enters a large valley, the horizon may actually be on the opposite crest. If the laser is pointed toward the horizon, it could miss an obstacle on the road at the bottom. However, the vehicle is also equipped with multiple fixed lasers that protect the vehicle. Future work will explore the addition of a maximal height filter to limit the degree of scan based on the maximum distance of its laser plane with the ground plane. This work could reduce the number of lasers needed on the front of the vehicle.

A drawback to a mechanical steering device is rapid transient response. Most terrain that the robot must traverse has a slow rate changes with respect to the Gnomon speed (full range deflection of -10 degrees to +10 degrees in about 250ms). However, as the vehicle travels over rougher terrain, the pitches and rolls can require compensation that exceeds the Gnomon's ability to react. Since the range gate produced by the vertical sensors is much quicker than the mechanical reaction time, it corrects for range errors introduced by the transient.

15.6.4 Down-Looking Lasers

While horizontal lasers are good at detecting large objects in front of the vehicle, they are not oriented to detect objects that lie close to the ground. Several teams addressed this issue by mounting horizontal lasers low to the ground where they are more susceptible to fouling and damage from low-lying obstacles and debris. Even so, horizontally mounted lasers are inadequate for detecting large holes or drop-offs in the terrain. Reports from the previous year (Urmson et al., 2004) and testing in the desert proved the need to detect these low-lying and negative

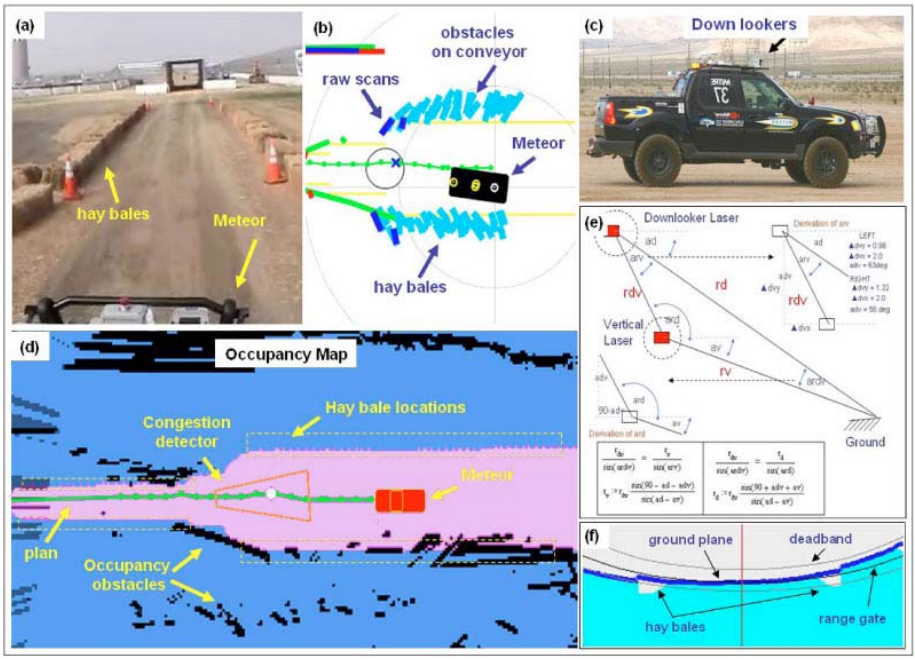

Fig. 15.11. Down looking. (a) Meteor travels down the center of low-lying hay bales. (b) Line segments generated by the down-looking lasers clearly show hay bales. Note Meteor correcting slightly right. (c) Down-looking lasers mounted to the top of the roof rack. (d) Occupancy map generated in the middle of the hay bales. The lane crosses the center image and narrows from right to left. Light areas indicate open space, dark segments are obstacles. Note that hay bales do not show up in the occupancy map. (e) Like the vertical lasers, the down-looker ground plane filter is based on the law of sines. (f) The raw output of a down-looking laser (shaded). The dark line riding on top of the raw readings is the horizontal ground plane. The hay bales are clearly seen in the scan. The ground plane shows that Meteor slowly rolled to the left.

obstacles. Roads in the mountains of the desert southwest are dominated by narrow paths with occasional 30-meter drop offs. They are also littered with large boulders (Figure 15.15d). Even in the relatively flat terrain of a lakebed, desert shrubbery can trap mounds of dirt.

To address these concerns, two additional lasers were mounted to the roof of the vehicle pitched down at 10 and 11 degree angles (Figure 15.11c). This down-looker scanned the road 10-15 meters in front of the vehicle. Using two scanners with slightly different angles increased the likelihood of seeing narrow obstacles in the road. The heightened roof position provided a better scan angle for sweeping across the ground. Lower positions would have required a shallower scan angle and would have made the scan distance highly dependent on vehicle pitch.

Like the horizontal lasers, the down-looking lasers rely on vertical lasers to provide an initial ground plane estimate for filtering out ground strikes. However, the vertical lasers only give an accurate sense of the terrain directly in front of the vehicle and do not accurately represent the terrain to either side. To overcome this limitation, the down-looking lasers employ their own rate change filter similar to the vertical lasers (Figure 15.11e). Both look at the change in slope of the ground and eliminate any ranges that change too quickly. This process is applied to the sweep starting at the center of the scan radiating out and uses the ground plane estimate of the verticals as a seed. This form of filtering is robust to the roll of the vehicle and rapidly changing terrain on either side.

A filter insensitive to terrain changes, pitch, and roll can be tuned very closely to the actual ground plane allowing detection of obstacles lying low to the ground. On relatively flat surfaces, such as parking lots and roads, the system can detect positive and negative obstacles as small as 6 inches above and below the ground plane. However, to be less sensitive to large vehicle perturbations (off-road terrain), the threshold was set to 10 inches. Vehicle ground clearance and large tires handle everything below that threshold.

Because the down-looking lasers point so sharply downward, they only see an obstacle as it passes in front of the vehicle once the scan moves past an obstacle, it is no longer visible. However, the vehicle still needs to account for its location to avoid steering into it. To maintain persistence, 3-dimensional line segments are generated to represent scanned obstacles and placed on a structure similar in concept to a conveyor belt.

Since the primary consumer of down-looker obstacles is the reactive system (discussed in Section 15.7.3), down-looker obstacles are represented in a relative coordinate system with respect to the front of the vehicle. As a consequence, this representation must be modified to account for movement. At each time step, obstacles placed on the conveyor belt are translated and rotated (about the nose of the vehicle) opposite to its motion to give their new positions relative to the vehicle. When obstacles clear the rear axle, they are removed from the conveyor belt.

Unlike objects generated by the horizontal lasers, once a down-looker passes over an obstacle, that obstacle is not sensed again. If the obstacle is sensed correctly, there is no issue. However, if the obstacle is incorrectly generated from noise or a ground strike, there is no mechanism for correcting it. The robot will try to avoid it and if no path is found, may stop completely.

To prevent oversensitivity to these kinds of failures, two mechanisms clear the conveyor belt. First, it is cleared whenever the robot is stopped. This assumes that obstacles at issue are small with respect to the path, and is consistent with the purpose of the reactive system. However, this means that if the robot were to approach a large drop-off across the road, it would first stop and then probably continue. This failure mode is outside the scope of the Grand Challenge.

The second mechanism is to react only to objects that are moving towards the vehicle head on. The down looker lasers scan directly in front of the vehicle

but also scan several meters to either side. If the vehicle is headed into a turn, this means that one side of the laser scan is running directly along the path instead of across it. As the vehicle then turns along that curve, it is looking at the previous obstacle from a poor perspective that may not clearly represent a true ground obstacle. To resolve this phenomenon, obstacles on the conveyor belt that must be rotated about the nose of the vehicle more than 45 degrees are removed.

The down-looking lasers proved themselves in the hay bale maze at the NQE. Just before the tunnel, two rows of hay bales about 50m long were placed on the either side of a narrowing lane. As the lane got closer to the tunnel, the lane boundaries shrank from 30m to 10m. Unless the GPS units were performing extremely well, it was difficult to travel down the center of the hay bale corridor without sensing them. As several teams discovered, it is extremely difficult to steer if a hay bale is lodged under a wheel.

Figure 15.11a shows the view from a camera mounted on top of Meteor as it enters the narrowing section of the hay bale maze just before reaching the tunnel. Figure 15.11f shows the raw output of the down-looker polar plot. The three parallel curved lines represent the uncorrected ground plane from the vertical laser and the dead band used to determine positive and negative obstacles. The dark line riding on top of the raw measurements is the lateral ground plane generated by the down-looker filter. The shape of the ground plane shows that the vehicle was rolling slightly to the left. Objects that exceeded 10 inches above this ground plane were treated as an obstacle. Depressions sensed below were treated as negative obstacles. In this plot, the hay bales on either side are evident whereas the occupancy map generated by the horizontal lasers does not show any hay bales (Figure 15.11d).

Figure 15.11b shows the down-looker obstacles that have been placed on the conveyor belt as the vehicle moves through the maze. The dotted lines show the boundaries defined by the route definition. The darker segments are obstacles from the current down-looker scan and the lighter segments are obstacles that are being retained and managed by the conveyor belt. Figure 15.11d shows the corresponding occupancy map generated by the horizontal lasers. Most hay bales do not show up in the occupancy map. Without the down-looking lasers, it is likely that the vehicle would not have stayed centered within the hay bale rows. As it turned out, the vehicle was able to navigate the hay bales without collision or incident.

15.7 Planning and Reactive Layers

Meteor employs a three-layer approach to maneuvering and navigation: waypoint following, path planning and hull protection. Waypoint following consists of mechanisms for determining speed and steering commands based on the defined route and was discussed in section 15.5. Path planning modifies the path of the vehicle based on a perceived obstacle field. Hull protection protects the hull of the vehicle whenever the path planning fails. These layers balance making progress down the road with avoiding collision and damage.

15.7.1 Occupancy Map

To effectively plan through an obstacle field, a robot must have a representation that shows obstacle locations and a safe passage between them. It must be able to accept and fuse information from sensors operating at different rates and restrictions. Moreover, since the vehicle is moving down a road, most of the sensors point forward and have a limited sensor footprint. Without a world representation, the vehicle may attempt to turn into objects that are present but no longer seen by the sensors.

Meteor uses an occupancy map a tessellation of space into equally sized grid cells. Each cell represents the likelihood that the corresponding space is either occupied or open on a continuous scale between 0 and 1. 0.0 represents open space with full confidence, 1.0 represents an obstacle with full confidence. 0.5 indicates no bias either way. Given no a priori information, cells of the map are initialized to 0.5. Figure 15.9d shows a typical occupancy map. The dark areas represent detected obstacles. The lighter areas represent open space. Uncertain space is the shade between.

Through a corresponding sensor model, each participating sensor can update cell probabilities based on how it perceives occupancy. For lasers, the sensor model is a triangular wedge that couples the defined line segment to the source of the laser that generated it. Two types of laser objects are fused into the map: line obstacles that represent solid objects and vistas that represent open space. Cells in the occupancy map that lie directly along the line segment of a line obstacle are updated to indicate a greater likelihood of being occupied. All cells that lie within the interior of the wedge are less likely to be occupied and are updated accordingly. A vista is similar to a line object except that only its interior is updated.

Occupancy maps have the ability to correct and arbitrate between inconsistent readings based on confidence. Sensors may erroneously report the presence of obstacles due to noise or an erroneous range gate. Those errors are projected onto the map to indicate impassible regions. However, new evidence from other readings is constantly fused into the map. As more correct readings appear, errors are reduced. If there is more confidence in a sensor reading, the correction occurs more quickly. Because the sensors report more false positives (obstacles that are not present) than false negatives (missing a real object), the sensor model is biased toward clearing the map more quickly than populating it.

To minimize processor load and complexity, a 2-dimensional occupancy map represents the terrain that the vehicle must pass through. Though lasers are 2-dimensional scans through a 3- dimensional space, their results can be projected onto a 2-dimensional map. However, each scans at a different angle and not all obstacles are the same height. As a result, a laser pointing slightly up might miss an object where another might see it. That puts the fusion mechanism in a dilemma since the confidence of both is high. This problem increases when one laser is updating at 75 times per second and the other at 5 times a second. The higher rate laser would dominate.

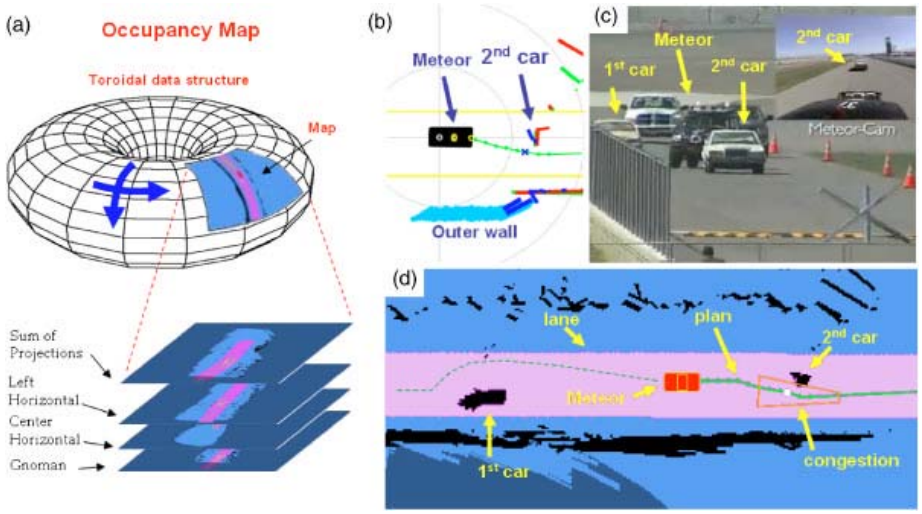

Fig. 15.12. Planning. (a) An occupancy map composed of multiple layers and summed. The structure is set up as a torus in that as Meteor moves, new information is added on one end and dropped from the other. (b) Laser information is converted to line segments and coded based on their source. (c) Meteor passes between two parked cars at the NQE. (d) The occupancy map produced while driving between the cars. The light areas are open space, the darker segments are obstacles, unknown space is the shade between. The route crosses through the center of the map. The line emanating from Meteor is its derived path around the car. The wedge is the congestion region. Note the two cars are shown in the lanes.

To solve this issue, projections are summed rather than projecting the sums. That is, the world is represented by multiple occupancy maps, one for each participating laser (Figure 15.12a bottom). Each laser builds its own map, fusing laser readings over time at its own rate. All maps are fused together 10 times per second to build a composite representation of the environment.

Another limitation of the occupancy map is the number of cells needed to represent an environment. The first year of the Grand Challenge spread over a large area and would have required a very large grid and memory footprint (Urmson et al., 2004). This storage problem was solved by building an occupancy map that acts like a moving window in space. As the vehicle moves, information behind the vehicle is shifted off the map and lost. Since the vehicle can shift in both x and y, the map can be envisioned as a large toroidal data structure (Figure 15.12a top). This representation has a fixed size and computational complexity. We chose 360 x 360 cells with each cell spanning 0.25m. That occupancy map provided a radial sensing distance of 45m. The dimensions of the map were a balance between processor load, memory footprint and the smallest size of obstacle it could represent. With these parameters, the occupancy map could be updated 10 times per second with a 30% processor load.

15.7.2 Route Projection

One of the difficulties of route following is deciding when a vehicle has reached one waypoint and should go towards the next. Because of obstacles or position error, a vehicle rarely reaches a waypoint exactly. When it recognizes this, it might circle back to try to get there exactly. Accommodations like expanding the radius of a point fail when waypoints are spaced closely together.

Since DARPA specified location and width of path positions, defining a lane, it was decided to optimize progress down the lane rather than determining whether Meteor reached a waypoint. Therefore, the route definition lane is projected and a plan is sought within it. This freedom eliminates many issues in deciding whether a waypoint was reached. Here the planner need only make continuous progress along the lane. Figure 15.12c shows an example of a lane projected into occupancy space.

A drawback to this method is that it does not guide the robot towards the center of the lane. On narrow segments, there is little room to wander, but in the event of a location offset, the method prevents the vehicle from trying to regain what it thinks is the center of a lane but is actually the edge. On wider segments, such as an open lakebed, the robot could wander several meters to either side of the route. Normally, this is not a concern because those routes tend to be wide open. To reduce wandering, projected lane widths were limited to 20 meters.

15.7.3 Planning and Verifying

Planning finds a viable path through occupancy space. The planner must select a route that not only avoids colliding with an obstacle but also respects the width of the vehicle and is resilient to changes.

Meteor employs an iterative greedy planner. The planner builds a plan in steps of 1.5 meters. At each planning step, it evaluates and selects the best next step from the end of the existing plan from choices fanned out in fixed angle increments. Each segment is verified by testing pixels adjacent to either side of the segment equivalent to the width of the vehicle along its length. If no obstacles are present, the segment is viable. For each planning iteration, the segment that makes the farthest progress down the lane is chosen. With no obstacles present, the selected segment is always the one parallel to the path. If the plan remains valid, it grows to the maximum range of the sensors: 40m. As the vehicle moves forward, the path shrinks and the planner builds it back up.

In addition to generating new segments, the planner also verifies the entire length of the existing plan 10 times per second. Though laser range finders can see beyond 30 meters, noise and geometry degrade the reliability of open and closed space calculations. As the vehicle gets closer to an object, the existing path might pass through an obstacle. Rather than invalidate the entire plan, the plan is truncated to a point just before the anomaly and is regrown from that point. The effect of this process looks like a snake constantly moving in front of the vehicle and around obstacles.

Figure 15.12d shows an example of planning through occupancy space. The wide line running down the center from left to right represents the lane defined by the route definition and restricts the planning space. The vehicle (center) is passing between the two parked cars in the later part of an NQE run (Figure 15.12c). The line extending from the vehicle toward the right is the current plan showing the individual planning segments. The two dark objects in the center of the lane (both front and back) are parked vehicles. Clearly, the planner has detected the second vehicle and has planned around it. The white dot along the path in front of the robot is the carrot. The thinner line extending behind the robot is the path the robot followed to this point and was added for illustration.

Finally, the planner provides a mechanism to feed congestion back to the speed regulator to slow the vehicle around obstacles. Testing showed that the vehicle moved up to a field of obstacles at full speed and attempted to quickly zigzag between them. Congestion represents the density of the obstacle field in the region in front of and to the sides of the current plan. It is measured by summing the number of obstacle pixels in a wedge-shaped region in front of the vehicle centered about the carrot. If the count exceeds a threshold, the robot is commanded to slow. The relationship was empirically determined through testing.

15.7.4 Reactive Approach

Maneuvering and navigation are primarily accomplished by planning a path through occupancy space. However, this has some drawbacks. First, fusion and integration of multiple sensor readings introduces latency in identification of obstacles. Several scans may be required for an object to reach the planner's obstacle threshold. In other cases, the object may lie on the inside of a turn and might not be seen until the vehicle is very close. This is especially bad if the space was reported as open or closed. Testing showed that an obstacle might not be detected until the vehicle is less than 10 meters away. In these cases, the planner would attempt to re-plan but the vehicle could not execute the new plan fast enough to avoid collision. One test site had a rapid drop immediately after one of the turns. Obstacles strategically placed just below this turn did not show up in the sensor sweeps until the vehicle had made the turn and was dropping into the depression. The vehicle often collided with these suddenly discovered obstacles. Though the elevation issue was solved by adding sensors, several cases remain where a similar phenomenon could occur.

A second complication results from heading and location error on the accurate integration of multiple sensor readings over space and time. When GPS is unreliable, a dead reckoning model is adequate to make vehicle progress but its fidelity is insufficient to maintain accuracy in the occupancy map. Laser measurements cannot be reliably correlated and so a blurred map and a less reliable plan result. This is especially true when the GPS signal does not completely drop out but degrades slowly over time.

Testing in desert terrain called for a more reactive mechanism to protect the nose of the vehicle. A purely reactive system does not rely on derived position or

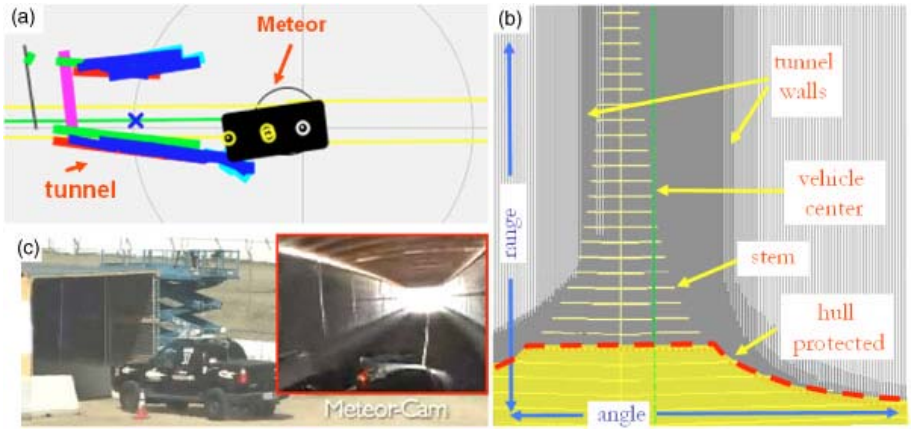

Fig. 15.13. Reactive. (a) Laser line obstacles generated as the vehicle passes through the tunnel. A mismatch in heading shows the vehicle thinking it is about to collide on the left. (b) Instantaneous lasers represented in polar coordinates with respect to vehicle centerline. The vehicle finds a steering correction angle that will guide the vehicle reactively and avoid obstacles. This representation shows the vehicle is closer to the right wall. (c) The vehicle as it enters the tunnel at the NQE. The tunnel is lined with metal sheets to block all GPS signals.

heading but instead creates its information directly from raw sensor data. Since the sensors themselves move with the vehicle, they provide a natural, vehicle centric-view of the world.

Meteor's reactive system, named Pelorus, receives inputs from the horizontal lasers and the down-lookers. The results are combined to produce a composite polar map of the open space in front of the vehicle. Pelorus is shown as an overlay of laser measurements plotted as angle vs. range in Figure 15.13b. The evenly-spaced, horizontal lines represent the projection of the vehicle in polar coordinates with respect to the front of the vehicle. This projection looks like the stem of an inverted wine glass. The shaded area represents the region Pelorus is protecting and moves up and down the stem as the speed of the vehicle changes. When the vehicle moves faster, the shaded region extends farther up the stem protecting a larger region in front of the vehicle.

If the space immediately in front of the vehicle is open, Pelorus is inactive and the planner controls. If an obstacle violates this space, Pelorus takes over and maneuvers the vehicle to avoid collision. When activated, a command is also passed to the speed controller to slow the vehicle. This shrinks the area that Pelorus protects and gives the vehicle more space to maneuver. If no viable openings can be found, the reactive system stops the vehicle.

The purpose of the reactive system is to prevent damage to the hull from obstacles that were not detected until too close to the vehicle. However, this reactive mode facilitates passing through a tunnel where GPS signals are lost. Figure 15.13 shows results from the reactive system as it passes through a tunnel

at the NQE during the third run. Figure 15.13a shows the vehicle about to collide with the tunnel wall on the left. Actually, the vehicle is traveling down the center of the tunnel (Figure 15.13c). The error is the result of a poor heading estimate as GPS degraded during entry into the tunnel. Figure 15.13b shows the instantaneous laser readings being reported from the reactive system in polar coordinates with respect to the center of the vehicle. These sensor readings do not rely on reported position and heading and show the tunnel is actually closer on the right and that the vehicle is well aligned. If the vehicle followed the plan, it would collide as it steered it farther to the right. Instead, the reactive system engaged to protect the vehicle and guided it the rest of the way through the tunnel. When the vehicle emerged, the GPS signal was recovered and the vehicle relied on planning again.

Competent navigation is a balance between directed motion based on a planning and dynamic steering correction based on a reactive protection system. Based on the terrain from last year's race, Meteor placed more weight on protecting the vehicle.

15.8 Developmental Approach and Field Testing

MITRE decided to enter the Grand Challenge in October of 2004. That left eleven months to create a vehicle that could operate autonomously in desert terrain. To achieve this goal required a solid plan, rigorous development and testing, and considerable discipline.

The team laid out a series of goals in one to two month intervals. Each of the goals involved a series of tests culminating in a demonstration. Engineers from outside the team were routinely invited to observe these demonstrations and assist with ideas and suggestions. Most testing was conducted in an abandoned parking lot (Figure 15.14a).

- November. Have a vehicle that could be driven under computer control. This was achieved by purchasing goods and services from Electronic Mobility Corporation.
- December. Drive autonomously from waypoint to waypoint. That was achieved with initial versions of a simulator, vehicle throttle and brake controllers, data loggers, data replayers, map display tools, and other software architectural components.
- January. Follow multiple waypoints with horizontal obstacle avoidance. By starting with a simple configuration [one GPS, one laser measurement sensor (LMS), and a laptop], end-to-end capability was demonstrated.
- March. Drive autonomously for five miles, using sensor data from multiple sources for obstacle avoidance while operating on varied terrain. Early speed and distance tests were performed at a farm in northern Virginia (Figure 15.14b). These long-distance tests forced resolution of sensor fusion, maneuvering near obstacles, advanced vehicle performance, and speed control.
- May. Prepare for a DARPA site visit. Testing had moved from obstacle avoidance to finding a balance between speed, planning, and reaction time. At this

point, sensing strategies were unresolved with stereo vision, radar, and machine vision for road detection under consideration. The site visit required completing a 200m course while avoiding placed obstacles (trashcans). The primary focus during this period was to minimize the number of trashcans we crushed. The site visit was successful (missing five of six trashcans).

- July. Take the complete system to the Nevada desert and test on the 2004 Grand Challenge route (Figure 15.14c). This was critical preparation for the race. Finding adequate, available test environments for regression testing was difficult. Emulating a desert in urban Virginia was a challenge. Out of ten months of testing, ten days were outside a parking lot. These ten days were extremely valuable and played a large part in getting to the finals.

  The desert provided a larger and longer testing space, permitting speed tuning in the flats of a wide-open lakebed and navigating trails bounded by large desert shrubs. It also provided an opportunity to test navigation through and around obstacles like cattle guards, switchbacks, and tunnels (Figure 15.14c). It demonstrated the hazards of the desert terrain including how radically and quickly the terrain could drop off and the size of the boulders (Figure 15.14d).

  July was also a self-imposed deadline for integrating new systems. While several months remained until the race, the systems needed to be rigorously tested and tuned to be successful. The sensor suite was finalized based on existing performance. Systems that were not sufficiently trusted were dropped.

  July saw evaluation of the effect of environment on sensors and processors. Heat was a concern but testing in 130-degree temperatures showed that the vehicle's air conditioning system was adequate to maintain cabin temperatures. Sensor fouling was also a concern. Desert dust is dry and fine-grained. It builds a small charge that causes it to stick to almost everything. Repeated experiments showed that the sensors worked without degradation, leading to the conclusion that this issue could be ignored.

The team traveled to the Yuma Proving Grounds in Arizona as a graduation exercise to test the entire system on an off-road test course (Figure 15.14e).

15.8.1 The National Qualifying Event

The National Qualifying Event (NQE) was the final DARPA evaluation before the big race. It was designed to test vehicle capabilities over a 10-day period at the California Speedway and to determine which vehicles should participate in the Challenge race. The course contained two large gates, several hills, hay bales, a tunnel, a debris field, a mock switchback, several parked cars, rumble strips, and a tank trap. Runs were evaluated based on the number of obstacles avoided and the time to complete the course. Each team had up to six attempts to complete the course — three complete runs were necessary to pass.

Meteor was one of the first teams to compete on day 1. It left the starting gate and headed for the first obstacle, two large, shiny gates positioned on either side of the route. Meteor made the turn, started to point between the gates, and

Fig. 15.14. Testing. a) Most vehicle testing occurred in an empty parking lot in McLean, Virginia. Here Meteor passes through a mock tunnel constructed from wood and weather tarps. b) Testing at local sites including a farm and an off-road facility. Note deep ruts that proved hazardous to low lying sensors. c) Six days of testing in 129 degree heat of Primm Nevada in July. Here Meteor successfully exits a tunnel and passes through cattle guards. d) Testing along the 2004 route included driving in rough mountain terrain. Note the size of the boulders left of Meteor. e) Testing on an off-road autonomous vehicle test course at the Yuma Proving Grounds a week before the race.

then veered away (Figure 15.15a) when DARPA stopped it. Figure 15.15b shows the Meteor view. Even though the gates appeared to lie slightly inside the route (probably due to GPS error), the vehicle should have been able to pass through them. After reviewing data logs, it was apparent that Pelorus did not think there was enough room for the vehicle to fit through the gate. The safety margin (1m) on either side of the vehicle was discovered to be too large. After folding the rear-view mirrors to reduce vehicle width, the safety margin was reduced to a few inches.

Run 2 did not go well. Meteor left the starting gate, traveled about 40 meters, and veered to the right. It continued its slow loop backwards and as it approached the outer barriers, DARPA officials stopped it. The system generated empty log files, which prevented analysis of the anomaly. Despite the lack of log data, the GPS multipath issue was discovered (see section 15.5.1) between runs 2 and 3.

Meteor entered run 3 with the same parameter set as Run 2. During this run, it passed through the first gates, traversed a grassy area including a slight incline and decline down to an asphalt road and into the hay maze. Figure 15.15e shows it navigating through the hay bales leading to the tunnel. Meteor entered the tunnel where it veered left and stopped after nudging the left tunnel wall. Log data showed that upon entering the tunnel the transition from GPS to motion model was not quick enough. Just before the motion model engaged, the GPS error produced a large heading discrepancy and Meteor veered left too quickly for Pelorus to react. Figure 15.15f shows the final resting place of Meteor. Though the angle is shallow, Pelorus did not think there was enough room to maneuver out of it. Replay of the data exposed the failure mode. Over the next few days, handoff between the GPS and motion model was tested. The team built a mock-up of a large tunnel. Figure 15.16g shows Meteor passing into the mock tunnel. GPS loss was induced by placing a ball cap covered with tin foil over the GPS unit, effectively killing its signal (the other GPS was turned off).

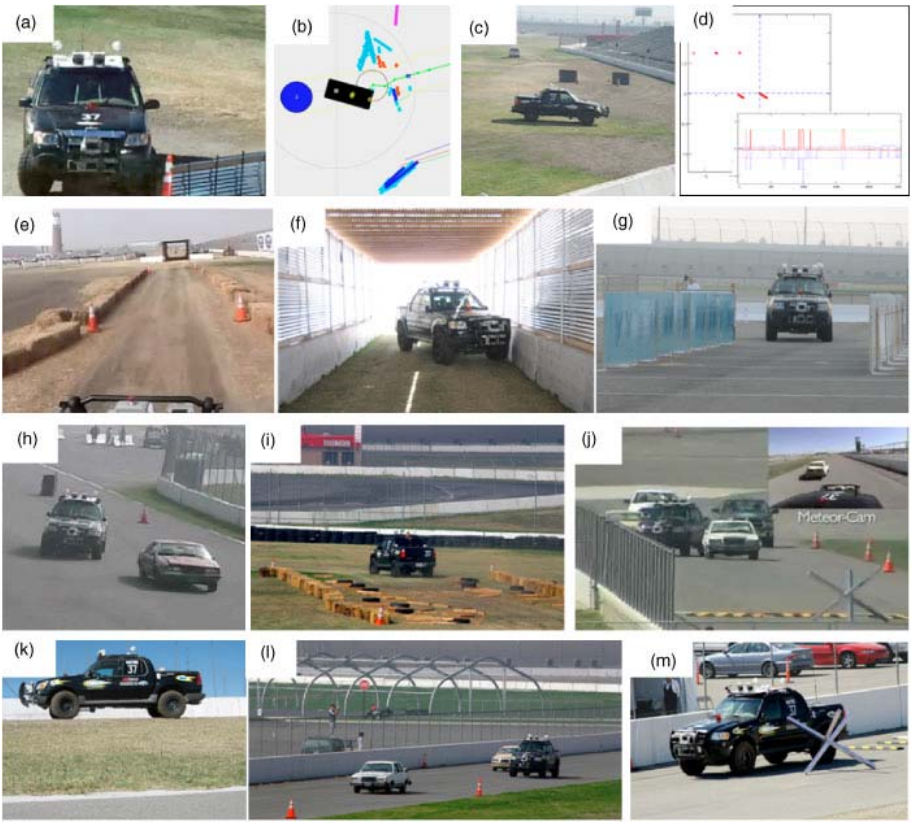

Fig. 15.15. NQE. Each team had six chances to complete the course. a,b) Meteor's first run ends as it fails to pass through the first gates. c,d) Meteor veers off course before reaching the gates. e,f,g) During the third run, Meteor clears the gate, rounds a steep hill and traverses a hay bale maze, but stops in the tunnel. Testing corrected this discrepancy. h-m) Meteor completes runs 4, 5 and 6 without incident, including avoiding several parked cars, a debris field, a mock mountain pass and a tank trap. Completing three runs qualified Meteor for the finals.

Testing showed the problem was the hand-off from planning and to reactive modes. Parameters were tweaked that make the GPS receiver threshold more sensitive causing transition to occur sooner and more often. The new parameters required a higher GPS confidence to stay in planning mode.

Once these issues were addressed, Meteor completed the next three runs with slow but repeatable times of 27 minutes each. Figures 15.15h-15.15m shows Meteor successfully avoiding obstacles at different parts of the 2.7-mile course. Figure 15.15h and 15.15j show Meteor driving around obstacles. Figure 15.15i shows Meteor driving over several small tires and 4x4s with no trouble. Figure 15.15k shows Meteor posed near the mock mountain pass. Barriers on the right

and a large drop off on the left simulated terrain found on a switch back. As with the hay bales, the down-lookers kept Meteor on the path driving between positive and negative obstacles. Figure 15.16m shows the last obstacle, rumble strips a few feet before a tank trap to see how vehicles sensed in rough terrain. In all runs, Meteor avoided the tank trap and crossed the finish line.

15.8.2 The Finals

Of the 152 teams that participated in the site visit and the 43 teams that competed in the NQE, only 23 qualified to race in the finals. The Meteor team was extremely pleased to compete in such a challenging event.

Because Meteor was slowest to complete the NQE, it was last to start the finals. As the day wore on, the winds grew stronger. By the time Meteor started, there was a steady breeze with periodic gusts. Meteor left the gate, traveled by the crowd and headed out to the desert. About thirty meters before entering the dry lakebed, it encountered a large dust cloud and stopped in a field of tall weeds and bushes.

During testing in the desert before the race, occasional dust clouds appeared as winds raised fine layers of desert dust. Laser scanners detected the translucent cloud and treated it as a transient obstacle. After the dust blew beyond the range of the lasers, the offending obstacle cleared and Meteor continued. Figure 15.16b shows a dust clouds as it passed Meteor from back to front in the race. Clouds were also generated by Meteor and the chase vehicle. Figure 15.16a is an overhead reconstruction of Meteor reacting to one of these clouds. The wide line running

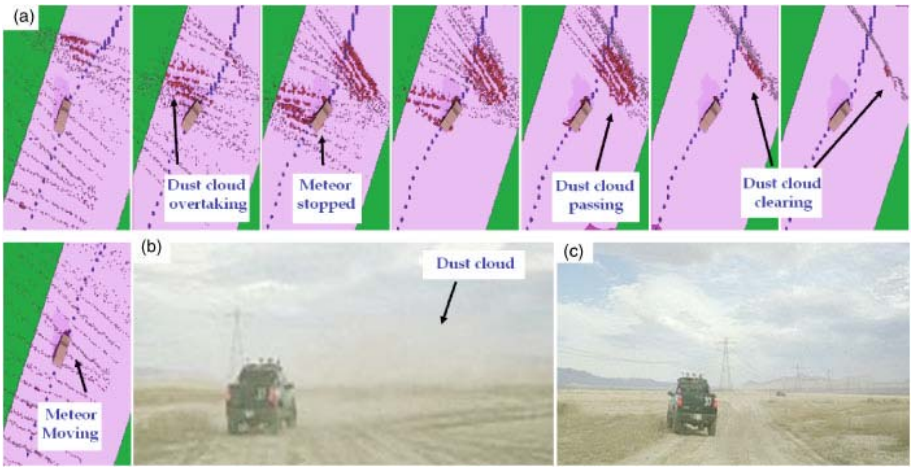

Fig. 15.16. The race. (a) Meteor approaches, slows, and attempts to avoid passing a dust cloud that periodically plagued it during its brief run. This sequence shows one of the many dust clouds as seen by the laser scanners. The thin lines are laser scans that hit the ground. The thicker lines are cause by dust. (b) A dust cloud passing in front of and away from the Meteor.

Fig. 15.17. The 2005 MITRE Meteor team. (Left to right) Frank Carr, Bob Bolling, Bob Grabowski, Richard Weatherly, Dave Smith, Ann Jones, Tiffani Horne, Mark Heslep, Keven Ring, Kevin Forbes, Mike Shadid, Laurel Riek, Alan Christiansen, and Sarah O'Donnell (inset).

through each segment is the lane defined by the route definition. Meteor is in the middle. The thin lines are laser scans that terminate at the range gate (hit the ground). The darker points are readings from perceived obstacles (Figure 15.16a middle).

The pictures start on the left where Meteor is moving on the course (as seen by the separation between laser scans). As the cloud reaches Meteor, it triggers the reactive system that forces it to veer and then stop. In most cases, as the dust clouds passed, Meteor would slow or stop and then continue along its path when the clouds cleared. This occurred 10 to 20 times during the run. The road started to turn left, so these clouds appeared to be coming from the right. The new angle pushed Meteor toward the outside of the road. Figure 15.16c shows one of these diversions. This occurred several times and ultimately pushed Meteor far enough off course that larger weeds and shrubs made the route look impassable. Since the lasers could not differentiate between weeds and large rocks, Meteor stopped just 1.1 miles from the starting gates and less than thirty meters from the wide-open expanses of the dry lakebed.

15.9 Conclusion

Although the MITRE Meteor team did not win the Grand Challenge, it did achieve its goals. MITRE's Grand Challenge experience validated three philosophies: Reliance on COTS, software location and employment transparency, and the model-build-test cycle. Purchasing COTS equipment and services focused project energy on autonomous robot control. Good examples are the installation of a COTS steering and propulsion servo control system and the vehicle suspension modification. Many teams did poorly at NQE when their homemade solutions failed.

A disciplined approach to software construction let good ideas accumulate without the system becoming fragile. One way to increase confidence in code is to employ it for as many purposes as appropriate. For the Grand Challenge, the

same body of code was used to operate the robot, run non-real time laboratory simulations, and study logs recorded in the field. This transparency of employment exposed problems in the lab before they could waste field-testing time.

The model-build-test cycle accumulated experience in a tangible and persistent way. When a problem was found or a new phenomenon identified, it was first modeled in the simulation environment. With a simulation of the problem or new phenomenon in hand, the body of operational code was adjusted to deal with it. Once proven in simulation, the robot was fieldtested to evaluate the changes and improvements were fed back to the model. A result of the model-build-test approach was that the model grew in fidelity and became a lasting repository of project experience. More importantly, this experience proved that testing in real environments is the key to success.

The experience gained by taking a vehicle to the Grand Challenge, though basic in nature, is a valuable asset to many Department of Defense and civilian missions. These lessons continue to equip MITRE to help our clients succeed in their robotic endeavors.

References

[1] Behringer, R., Sundareswaran, S., Gregory, B., Elsley, R., Addison, B., Guthmiller, W. (2004, May). The DARPA Grand Challenge - Development of an Autonomous Vehicle. Paper presented at the IEEE Intelligent Vehicles Symposium, Parma Italy.

[2] Cheok, K.C., Smid, G.E., Kobayashi, K., Overholt, J.L., Lescoe, P. (1997, June). A Fuzzy Logic Intelligent Control System Architecture for an Autonomous Leader-Following Vehicle. Paper presented at the American Control Conference, Albuquerque, New Mexico.

[3] Crane, C. III, Armstrong, D. Jr., Torrie, M., Gray, S. (2005). Autonomous Ground Vehicle Technologies Applied to the DARPA Grand Challenge. Paper presented at the International Conference on Control, Automation and Systems, Bangkok, Thailand.

[4] Fulton, J., Pransky, J. (2004) DARPA Grand Challenge A Pioneering Event for Autonomous Robotic Ground Vehicles. *Industrial Robot: An International Journal* Vol. 31, No. 5, 414–422.

[5] Gillula, J. (2005). Data Fusion from Multiple Sensors: Real Time Mapping on an Unmanned Ground Vehicle. Summer Undergraduate Research Fellowship Final SURF report.

[6] Kuhl, F., Weatherly, R., Dahmann J. (2000). *Creating Computer Simulation Systems: An Introduction to the High Level Architecture*. Prentice Hall PTR.

[7] Manduchi, R., Castano, A., Talukder, A., Matthies, L (2003, September). Obstacle Detection and Terrain Classification for Autonomous Off-road Navigation. *Autonomous Robot*, Volume 18.

[8] Minsky M. (1985). *The Society of Mind*. Simon and Schuster.

[9] Ozguner, U., Redmill K., Broggi A. (2004, June). Team TerraMax and the DARPA Grand Challenge: A General Overview. Paper presented at the IEEE Intelligent Vehicles Symposium, Parma Italy.

[10] Saptharishi, M., Bhat, K.S., Diehl, C.P., Dolan, J.M., Khosla, P.K. (2000, September). CyberScout: Distributed Agents for Autonomous Reconnaissance and Surveillance. Paper presented at the Conference on Mechatronics and Machine Vision in Practice.

[11] Sato, N., Matsuno, F., Shiroma, N., Yamaski, T., Kamegawa, T., Igarashi, H (2004, September). Cooperative Task Execution by a Multiple Robot Team and Its Operators in Search and Rescue Operations. Paper presented at the IEEE/RSJ International Conference on Intelligent Robots and Systems, Sendai, Japan.

[12] Sotelo, M.A., Rodriguez, F.J., Magdalen L. (2004). VIRTUOUS: Vision-Based Road Transportation for Unmanned Operation on Urban-Like Scenarios. *IEEE Transactions on Intelligent Transportation Systems*, Vol. 5, No. 2.

[13] Urmson, C., Anhalt, J., Clark, M., Galatali, T., Gonzalez, J., Gowdy, J., Gutierrez, A., Harbaugh, S., Johnson, M., Roberson, Kato, Y., Koon, P., Peterson, K., Smith, B., Spiker, S., Tryzelaar, E., Whittaker, W. (2004). High Speed Navigation of Unrehearsed Terrain: Red Team Technology for Grand Challenge 2004. (Tech. Rep. CMU-RI-TR-04-37). Pittsburg PA: Carnegie Mellon University, Robotics Institute.

[14] Woolridge, M., Jennings, N., Kinny, D. (2000). .The Gaia Methodology for Agent-Oriented Analysis and Design. *Autonomous Agents and Multi-Agent Systems*, Volume 3, Issue 3, 285–312.