Stanford's Robotic Vehicle "Junior:" Interim Report

Stanford Racing Team, Stanford University, Stanford, CA 94305, USA

April 12, 2007

***Abstract*—This interim report describes the vehicle "Junior," which is the Stanford Racing Team entry into the DARPA Urban Challenge. We survey the current state of the hardware and software development, and discuss ongoing experiments.**

I. Introduction

THE Stanford Racing Team won the 2005 DARPA Grand Challenge, using a modified VW Touareg dubbed "Stanley." For the 2007 Urban Challenge, Stanford has been selected as Track A Participant.

This article serves as an interim report for the team's progress, in partial fulfillment of DARPA's requirements of the Track A Urban Challenge Program. The report describes the existing hardware and software components, and lays out the ongoing evaluation and development plan.

II. Team Composition

The Stanford Racing Team is comprised of students, staff, and faculty of Stanford University and various affiliated organizations. The team composition is largely identical to the one that developed Stanley in 2005. The members of the Stanford Racing Team are drawn from the following academic and corporate entities:

- Stanford University (lead responsibility for software development and overall project lead)
- Volkswagen of America, Electronics Research Lab (lead responsibility for vehicle development)
- Mohr Davidow Ventures (lead responsibility for communications and outreach)
- NXP (founded by Philips)
- Google
- Intel
- RedBull

More information about the Stanford Racing Team can be found at www.stanfordracing.org.

The overall team lead is Sebastian Thrun (Stanford). The vehicle development is lead by Burkhard Huhnke together with Ganymed Stanek and Suhrid Bhat (all from VW ERL). The software development is lead by Mike Montemerlo, with Jesse Levinson, Anya Petrovskaya, Gabe Hoffmann, Doug Johnston, and Dirk Hähnel (all of Stanford University), and Dmitri Dolgov (Toyota Technology Center). Finally, the communications lead is Pamela Mahoney (MDV) with David Orenstein (Stanford University) and Steve Keyes (VW). Approximately 20 other students and staff members are working on various aspects of the software and hardware. Most team members work full time on the project. Some team members initially participated in the development through the Stanford course *CS294-Projects in Artificial Intelligence*, which was taught in the Winter Quarter of AY 2006/07.

Team meetings take place weekly for the technical team, with various subgroups meeting more frequently. Meetings of the advisory board take place at Stanford once a month.

III. Vehicle Platform

A. Vehicle and Instrumentation

Junior is based on a stock VW Diesel Passat Wagon, as presently sold in Europe. In total, our development utilizes three vehicles, where one serves the role of the primary race vehicle, and the two other vehicles are used as backup and for development purposes. The following image shows Junior (this image is a photo-illustration; the present development vehicle is not "stickered" as shown below):

For development, one of the vehicles has been modified for computer control; the other two vehicles are presently being modified. A low-torque DC motor is connected by a belt to the steering column to enable autonomous steering. Direct electronic interfaces into the brake booster, throttle, gearbox, and turn signals allow for tight control of the vehicle's speed and direction. Additional interfaces supply the onboard computers with vehicle state data, such as steering angle and individual wheel speeds. A wireless kill switch and physical e-stop buttons are presently being integrated.

The development vehicle also possesses a battery backup system, which can be plugged into an external power supply for keeping the computers running while the vehicle's engine is shut down.

The trunk assembly is visible in this photograph:

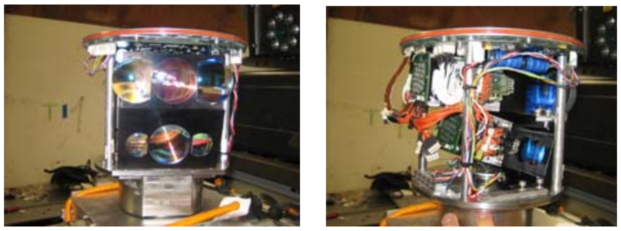
B. Computers

All control computers are mounted in the trunk of the vehicle. Presently, this comprises two PCs with Intel multi-core CPUs, and interface control modules for the various sensors and actuators. All computers run Redhat Linux - FC6.

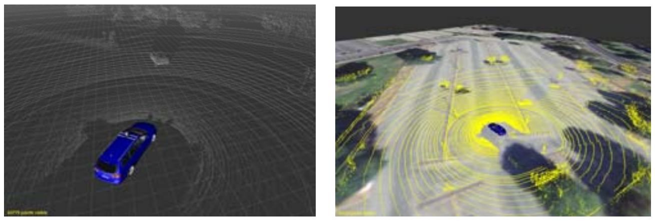
C. Sensors

Junior uses the Applanix POS LV 420 Navigation system for state estimation (location, orientation, velocities). The POS LV 420 system comes with three GPS antennae, mounted on the roof of the vehicle, a high quality Inertial Measurement Unit, mounted in the trunk over the rear axle, and an external wheel encoder, attached to the left rear wheel.

For external sensing, Junior presently features a Velodyne HD LIDAR laser range finder, shown here without the actual enclosure:

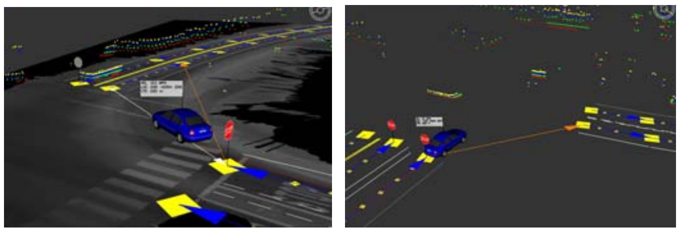


The following images show typical range scans, acquired by this scanner: The data is obtained in full 3-D, and geo-registered using the Applanix pose information.



Additional range sensing is achieved through two IBEO Alasca XT sensors, mounted on the front bumper of the vehicle.

The following images show typical IBEO range scans:



The IBEO data is sparser than the Velodyne HD LIDAR, yet the ground is filtered out automatically, making the data easier to process.

Additionally, Junior uses an omni-directional Ladybug camera, manufactured by PointGray. This camera is comprised of six CMOS video cameras, connected to Junior's computers through a firewire interface. The following figure shows an example snapshot of multiple cameras:

The cameras are calibrated and images are projected onto a spherical model, to provide full panoramic imaging, here superimposed on an aerial map of the environment.

This sphere, from the vehicle's viewpoint, provides a full omni-directional view of the area surrounding the vehicle, which also includes the road surface.

The present system also possesses two long-range radar sensor manufactured by Bosch. These sensors provide additional range data of vehicles, complementing the laser data. Radar is used to detect vehicles at extended ranges, and vehicles that are occluded by other vehicles.

In the near future, two SICK LD-LRS1000 lasers will be mounted on the back corners of the vehicle, to ensure complete coverage behind the vehicle for reverse driving and passing maneuvers.

IV. Perception Software

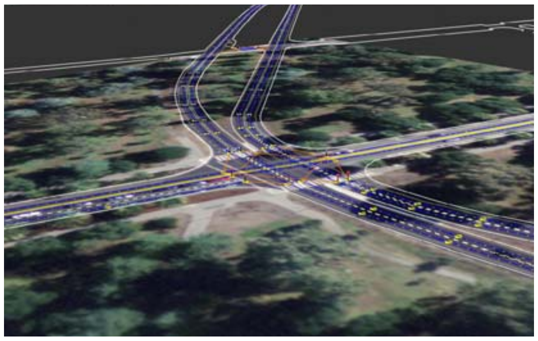
A. *Software Platform*

Junior inherits from Stanley a distributed modular software architecture, through which dozens of modules process and propagate data asynchronously. The architecture uses Simmons's IPC (Inter-Process Communication) software for communication.

Incoming data from the robot and its sensors are pipelined through multiple stages, comprising: sensor interfaces, perception and state estimation, planning, control, and vehicle interfaces. The pipeline is executed in parallel on all processors, imposing a total processing delay of approximately 300ms between sensor measurements all the way to vehicle control.

The software also logs all data, and integrates the sensor data with other data sources, such as aerial imagery.

Another key facility of Junior's software is visualization. A rich visualization suite makes it possible to monitor the robot's state and spot problems. The following image depicts an RNDF file of campus superimposed on aerial imagery obtained from an online source.

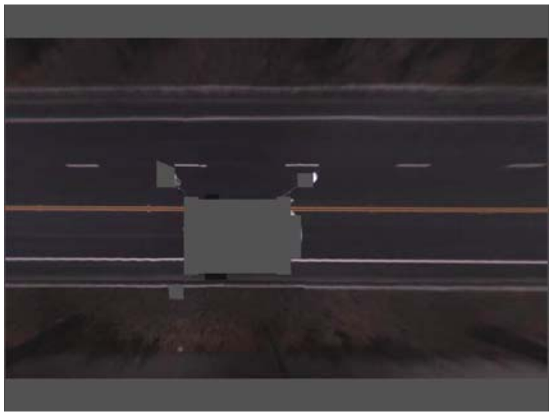


For the Urban Challenge, we developed a new suite of software for reading in RNDF files and tracking the vehicle relative to these files.

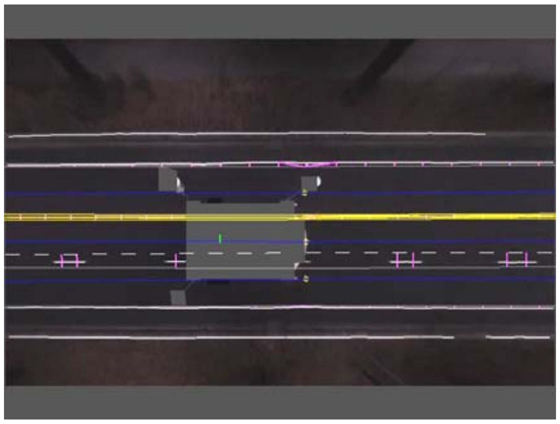
B. *RNDF Localization*

The primary function of the panoramic camera system is lane marker detection, and precision localization. The Stanford Racing Team has developed a full processing pipeline for lane marker finding in images, and precision alignment relative to the RNDF. This pipeline operates in real-time, using a special on-board Graphical Processing Unit (GPU).

In the first processing step, Junior's pipeline corrects for roll and pitch of the vehicle, and then "rectifies" the panoramic image into a flat overhead image. In this image, lane markers are now in the same geometric reference frame as the lane information in the RNDF, using the same relative measurement units:

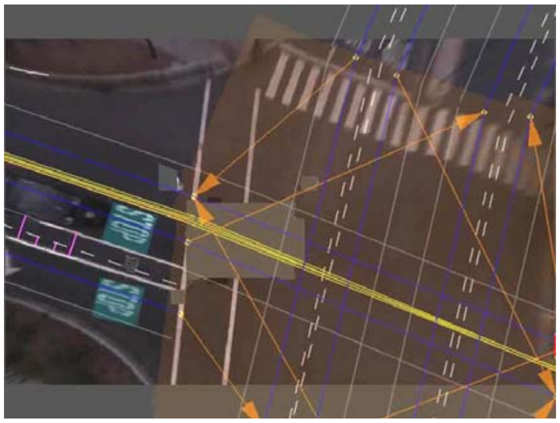


The image is then analyzed for the location and color of lane markers. A least squares process aligns the found lane markers with the RNDF, correcting for possible GPS errors.



This “aligned” image shows the RNDF superimposed to the lines found by our algorithm. Small pink bars visualize the residual error in this process. The precision of this alignment is usually within a few centimeters.

To increase the robustness of this approach, our method automatically rejects regions of high ground clutter, as well as intersection regions. Examples of rejected regions include words on the pavement (shaded cyan), and intersections (shaded orange) in the image below. This is all achieved without any human intervention.

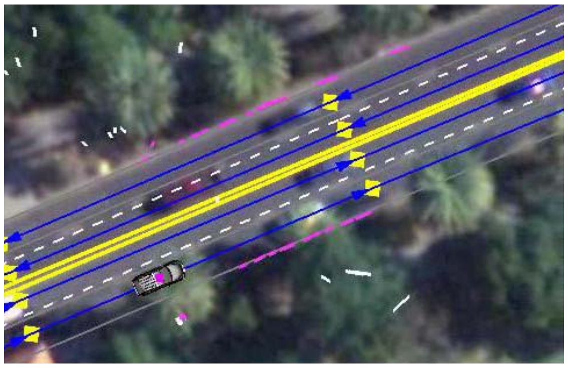


The result of this alignment process is fed back into the pose estimation module, to further refine the position estimates of the GPS system. As a result, Junior knows its lateral location on a lane usually with an accuracy of five centimeters. The longitudinal accuracy is typically half a meter, due to the lack of environment variation in the driving direction. The process runs online.

C. Curb Finding

Junior also analyses Velodyne range data to identify small, curb-like obstacles. These obstacles are found through an analysis of individual scan lines. In Junior’s software, a machine learning method has been trained to detect “curb signatures.” Once a sub-like structure has been found, it is used in collision avoidance and precision localization.

The following image illustrates curb finding for a single range sensor scan. In this image, curbs are colored white and pink. White curbs are spurious (e.g., tree trunks); pink curbs have been matched to the RNDF. As can be seen in this image, Junior find curbs reliably, and all spurious curbs are only found outside the RNDF lane area, where they do not impact Junior’s driving performance:



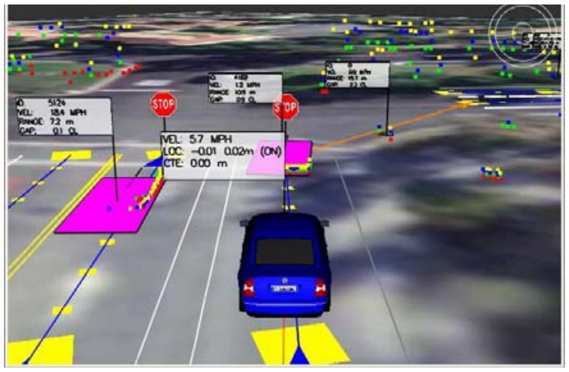
The curb finder operates in real-time. Curb data will be used both for collision avoidance and for localization.

D. Vehicle Tracking

Special routines have been developed for tracking moving objects in the environment. The present tracking method can independently use range data from the Velodyne and the IBEO range sensors, and it utilizes radar data to augment the tracking results.

The processing pipeline for object tracking proceeds in multiple steps. First, the range data is filtered for vertical obstacles, and areas outside the RNDF are discarded from consideration, as are measurements of the ground plane.

Next, the processing pipeline uses particle filters to fit 2-D rectangles to the surviving data (the z-dimension is ignored in this analysis). Each rectangle may be moving or stationary, and multiple rectangles may be necessary to explain a sensor scan. For moving objects, the tracked rectangles are annotated with an estimated velocity. The following image illustrates a typical situation with multiple moving objects.



The outcome of the dynamic object tracker is a list of tracked objects, annotated by their velocity. Stationary objects are specially marked, since they might require different behavior. In empirical testing, this approach was found to be highly accurate and reliable.

E. RNDF Construction

For training purposes, the Stanford Racing Team has constructed a method for semi-automatically generating RNDF files. This method analyzes logged laser data for the location of lane markers, and it uses Markov Random Field techniques for lane marker completion when lane markers are locally washed out. The following two examples illustrate the methods when applied to highway data:

Automated Lane Extraction from Laser Intensity

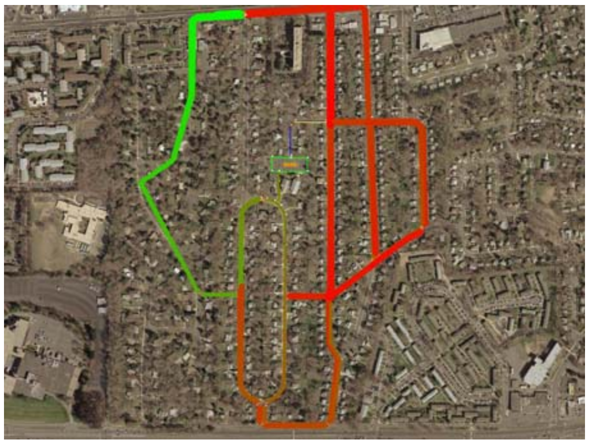
Automated Lane Extraction from Laser Intensity

Clearly, this method will not be useful for the final race; yet it enables the team to quickly acquire new RNDFs in new training environments.

V. Planning and Control Software

A. Global Route Planning

The route planer uses dynamic programming to propagate a cost function over the entire road network defined by the RNDF. This function estimates the expected time it will take to reach a given goal location, from any location in the world. Am example using the DARPA-provided sample RNDF is shown here:



The cost of each segment is computed from an analysis that determines the right of way at each intersection, and penalizes potential yield times, and slow-downs in turns.

It is important to note that the fact that this analysis is performed for each individual lane of a multi-lane road segment. By asserting a particular a priori probability of failure for lane change operations, the resulting cost function penalizes being in a left lane close to an intersection where a right turn is expected. Given no other data, this cost function describes the minimum cost to the goal from every location in the RNDF. When used in conjunction with local sensor data, the cost function can be used to evaluate the relative costs of alternative routes to the goal, allowing the robot to immediately replan routes given slow traffic conditions or road blockages.

The dynamic programming is rerun each time the robot's goal location is changed (once for each MDF waypoint). The value function can be computed in less than 100ms for RNDFs far larger than the sample RNDF, making it feasible to start driving without any noticeable delay.

B. Online Path Planning

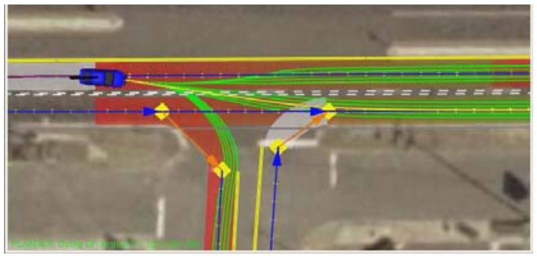
A key component of the driving system is Junior's online path planner. The path planner is responsible for tactical decisions such as lane changes, merging, and avoiding local obstacles. Its planning horizon is derived from the (minimum) maximum range of the range sensors.

The path planner searches for appropriate paths along two dimensions:

1. Discrete choices, such as lane changes and turns (the "macro-plan"), and
2. Continuous choices, such as the specific lateral offset within a lane (the "micro-plan").

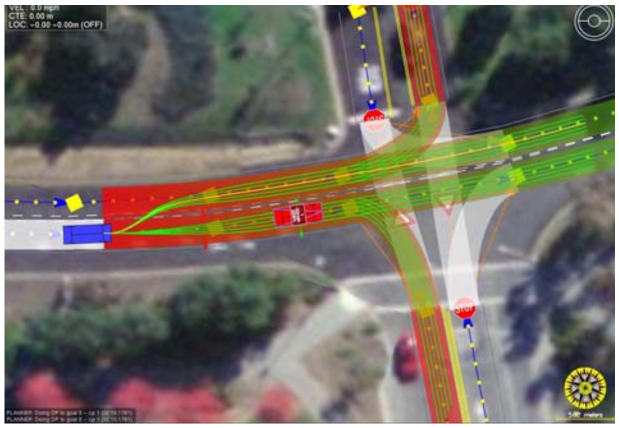
Each pair of macro- and micro-plan is evaluated using three scores: a local cost, a global cost, and a maneuver cost. The local cost evaluates the proposed vehicle trajectory given the local configuration of obstacles. This cost estimates how fast this trajectory can be driven relative to its maximum speed limit. The global cost is the minimum value of the value function along the macro plan, as obtained via the dynamic programming method just described. This estimates the time required to reach the goal from the end of the local trajectory. Finally, the maneuver cost encodes the fact that certain maneuvers (such as lane changes and u-turns) are dangerous and thus should incur extra cost. For example, the vehicle should only change lanes if the difference in path cost exceeds the maneuver cost. Together, these criteria maximize progress while yielding stable, non-oscillatory behavior.

The following image illustrates, in green, legal paths considered by the path planner.

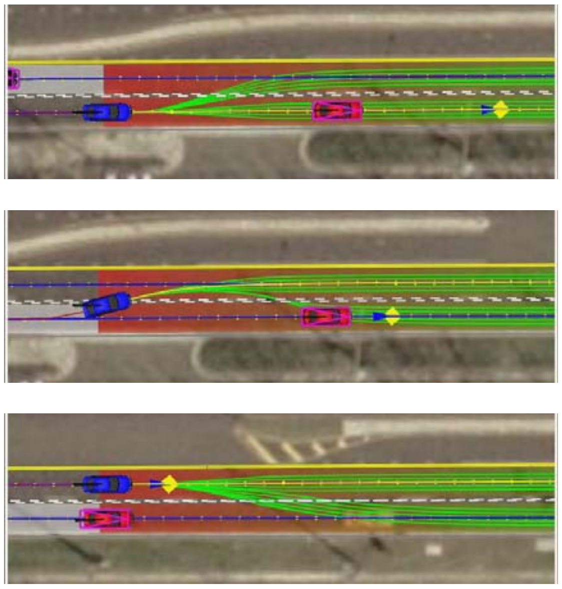


Those trajectories are generated automatically from the local RNDF, with discrete changes of lateral offset relative to the lane center.

Complex RNDFs can make for complex maneuvers. The present planner can accommodate arbitrary geometries and topologies of the road network, as illustrated in the following illustration. In particular, the robot only considers left or right turns when in the appropriate lanes. Thus, to turn left, it first needs to change lanes.



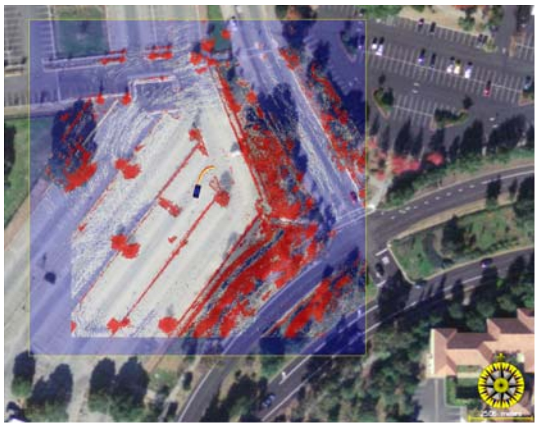
The path planner considers objects found by Junior's object tracker. The treatment for moving and static obstacles differs. Junior never considers swerving around moving obstacles; instead, its default response is to slow down. When facing a non-moving object, however, Junior modifies its cost function so as to favor paths that pass the obstacle. The following sequence of images illustrates a successful lane change maneuver for passing a stationary obstacle. It is important to note that while certain action choices, such as lane changes, are discrete in nature, the planner can modify these decisions at any point in time. As a result, lane changes can be aborted immediately if a previously unseen hazard is revealed. These contingency plans can be seen in the second lane change image.



Once a particular trajectory has benen chosen, Junior assigns to it the maximum permissible velocity. This velocity is calculated as the minimum of the MDF speed limits, constraints that arise from the curvature of the trajectory, and velocity that arises from maintaining a safety margin to other objects. For each plan, Junior can construct all possible RNDF paths that merge or cross its planned path, respecting the rules of right-of-way. The vehicle speed is set so that safe margins are maintained with other traffic that is allowed to merge or cross the given trajectory. These merging and crossing zones are depicted in the previous images shaded in white.

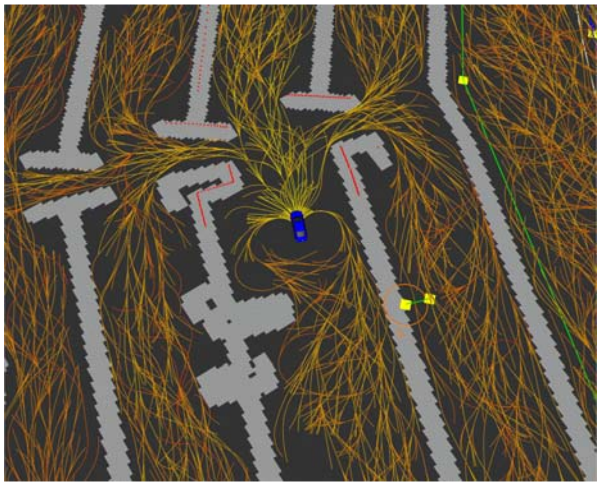
C. *Parking Lot Navigation*

For parking lot navigation, the Stanford Racing Team has developed a multi-layer path planner that can generate unconstrained paths. These paths do not follow an RNDF, and they can involve multiple turns. The path planner operates fast enough to adapt the path to momentary range measurements, and past measurements are cached into a map so that obstacles are considered even when outside the momentary field of view of the vehicle:

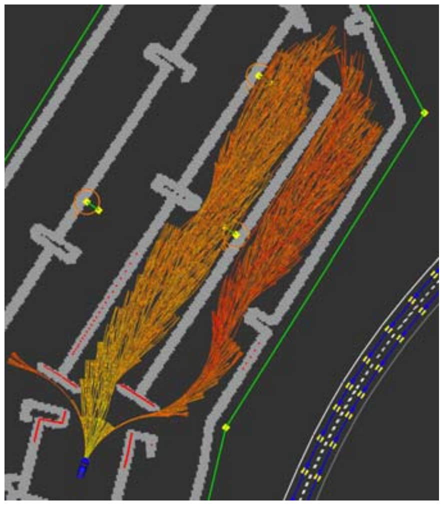


The blue area in the image corresponds to the local map, which is superimposed on aerial imagery.

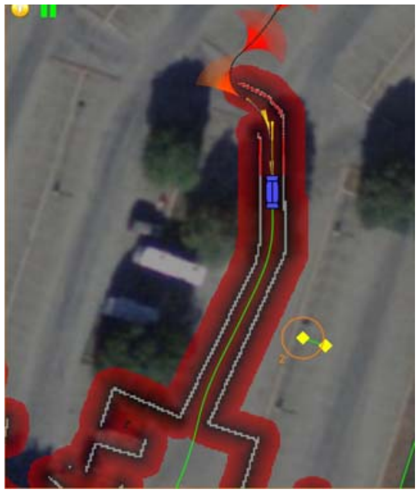
The parking lot path planner uses a combination of 3-D dynamic programming on a coarse grid, and the Rapidly-Expanding Random Tree (RRT) algorithm. Trajectories are generated using a physical vehicle model with infinite tire stiffness. The breadth of possible paths is illustrated by the following example:



To guide this search, the value function in determining the goodness of a path is computed based on the distance to the goal (e.g., a parking spot), path smoothness, and clearance to obstacles. Under development is an extension that enables Junior to back up and drive backwards. The following image shows the directed search, in which the search tree is grown in response to a value function generated by the initial discrete value function calculation:



A difficult situation is depicted in the following image, in which Junior navigates between two tight rows of obstacles.

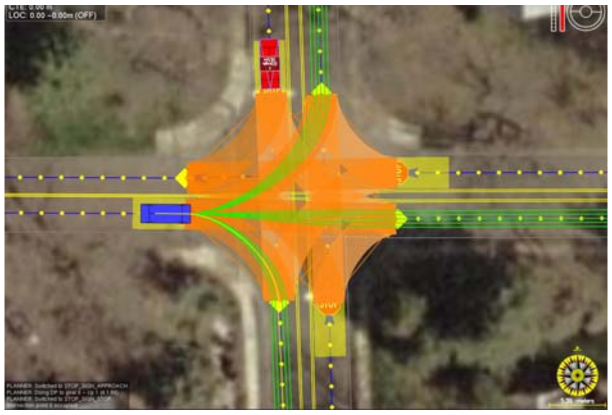


D. Control

Junoir's controller is adapted from Stanley, but has been enhanced significantly to directly control steering torque, instead of just steering angle (as was the case for Stanley). The controller has been tuned using *reinforcement learning* methods,. In experiments on an open airfield, Junior has successfully traveled at speeds up to 71mph.

E. Stop signs and intersections

To handle stop-signs and intersection, Junior analyzes the RNDF for intersection regions. According to the rules of the challenge, only one moving object is allowed inside the intersection at a time. The following diagram illustrates a typical marking of an intersection. Shown in orange here is a region that is marked for exclusive use.



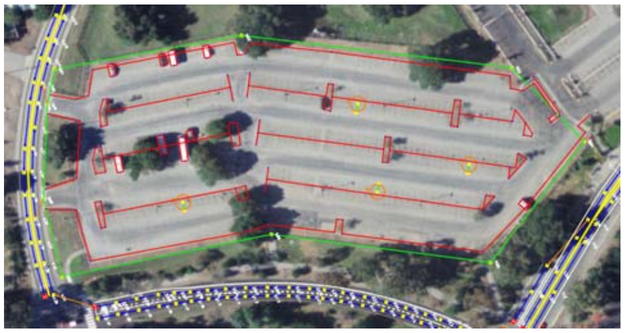
The actual logic for handling intersections combines a state machine that track the arrival and departure of vehicles that arrived first at the intersection, paired with a timer that overrides the state of waiting after a certain amount of time. To break ties, the wait time is randomized. In extensive simulations, we found randomized wait times to be uniformly superior to deterministic wait times, which can lead to collisions if two vehicles use identical timeouts.

VI. Safety and Development Support

The current robot system features a system that enables a human driver to seamlessly switch between robotic and manual driving, simply by tapping the brake or by asserting a slight force on the steering wheel. A special software module monitors the health status of the software system and alerts the driver using the onboard stereo system, in case of possible irregularities. While such a system is not necessary for the final race, it is essential for driver and vehicle safety in the development phase.

VII. State of Experimentation

Some of the experimentation is carried out in simulation, or in combination of simulated and physical data. To this end, the Stanford Racing team has developed a multi-vehicle simulator that can be configured for arbitrary environment geometries and RNDFs. An example of a simulated parking lot is shown here—this specific example is derived from an actual parking lot, as indicated by the underlying aerial image.



In simulation, we have been able to identify a number of possible failure modes. In particular, our Urban Challenge Simulator was used in Stanford CS224M Multi-agent Systems, a graduate course taught by Prof. Yoav Shoham. Students in this course were given the task of developing vehicle controllers to compete in a final competition—all based on what they learned about game theory in this class. Probably the most import outcome was that most vehicles quickly became stuck indefinitely, just by unanticipated reactions that departed from "typical" human driving. We are presently exploring the development of robust driving strategies that can cope even with misbehaving robots, while performing according to normal driving conventions and rules.

Over the past months, much of our testing has involved physical vehicle operation. Vehicle testing, at this point, is carried out about once every week at Moffett Field. NASA Ames, who administers this air field, has graciously agreed to our use of their air field. Moffett Field possesses two runways, both approximately 2 miles long. The following image was taken during an experiment on Moffett Air Field:

Stanford Campus is also used for physical vehicle experiments. In all experiments, the vehicle is staffed with a safety driver who can take over control simply by grabbing the steering wheel, and a technical officer who monitors the vehicle's software. On Stanford Campus, Junior has passed a sequence of driving tests. The most difficult of those tests involved a 10-mile experiment in which Junior had to drive in traffic, line up at an intersection, stop at stop signs, and take several turns. All of these maneuvers were driven autonomously with one exception: the "resume" after a stop sign was triggered manually, for safety. Stanley passed this test, and exhibited robust localization, vehicle tracking planning, and control. Compared to the Stanley progress at this point in time, Junior is far ahead.

VIII. Outlook

At present, the Stanford Racing Team is in its final push to bring together the end-to-end system with *all* necessary behaviors and capabilities. We anticipate that this important milestone will be achieved by April 30, 2007. From this point on, the Stanford Racing Team will freeze all working hardware and software modules, and engage in elaborate testing. Testing will commence throughout the summer, similar to the development path that led Stanley to success in the 2005 DARPA Grand Challenge.

IX. Acknowledgement

The Stanford racing Team is indebted to DARPA for creating the Urban Challenge, and for its financial support under the Track A Program. Further, Stanford University thanks its various sponsors. Special thanks also to NASA Ames for permission to use their air field.