AI Magazine Volume 27 Number 4 (2006) (© AAAI)

A Personal Account of the Development of Stanley, the Robot That Won the DARPA Grand Challenge

Sebastian Thrun

■ This article is my personal account on the work at Stanford on Stanley, the winning robot in the DARPA Grand Challenge. Between July 2004 and October 2005, my then-postdoc Michael Montemerlo and I led a team of students, engineers, and professionals with the single vision of claiming one of the most prestigious trophies in the field of robotics: the DARPA Grand Challenge (DARPA 2004).[1] The Grand Challenge, organized by the U.S. government, was unprecedented in the nation's history. It was the first time that the U.S. Congress had appropriated a cash price for advancing technological innovation. My team won this prize, competing with some 194 other teams. Stanley was the fastest of five robotic vehicles that, on October 8, 2005, successfully navigated a 131.6-mile-long course through California's Mojave Desert.

This essay is not about the technology behind our success; for that I refer the interested reader to recent articles on the technical aspects of Stanley (Dahlkamp et al. 2006; Montemerlo et al. 2006; Stavens and Thrun 2006; Thrun, Montemerlo, and Aron 2006; Thrun et al. 2006). Instead, this is my personal story of leading the Stanford Racing Team. It is the story of a team of people who built an autonomous robot in record time. It is also a success story for the field of artificial intelligence, as Stanley used some state of the art AI methods in areas such as probabilistic inference, machine learning, and computer vision. Of course, it is also the story of a step towards a technology that, one day, might fundamentally change our lives.

Thinking about It

My story begins in March 2004. Both Michael Montemerlo and I attended the first Grand Challenge Qualification Event at the Fontana Speedway, and Mike stayed on to watch the race. The race was short: within the first seven miles, all robotic vehicles had become stuck. The best-performing robot, a modified Humvee dubbed Sandstorm by its creators from Carnegie Mellon University (CMU), failed after driving just 5 percent of the course (Urmson et al. 2004). Even though I had never worked on autonomous cars, I could not help but to ask the obvious question: could we do better? And, more importantly, why was the Grand Challenge hard? Why did so many teams falter in the first few miles? And what could we learn by getting involved?

In July 2004, my research group decided to give it a try. Word of our decision traveled fast. Within two weeks of our initial decision to participate, Cedric Dupont from Volkswagen (VW) of America contacted us, offering his lab's support for a Stanford entry in the race. Cedric worked in a lab called Electronics Research Lab headed by Carlo Rummel, and located only two miles from campus. VW was in the process of marketing its new Touareg SUV in the United States. The lab saw the race as an opportunity to showcase its Touareg in a high-profile event, while working with us on new car-related technologies. Cedric offered our team two VW Touaregs equipped with drive-by-wire systems, plus full engineering support. This offer was irresistible. We were to have a robot car, one that was highly suitable for desert driving!

Before we received our robot, VW lent us a conventional Touareg. One of the very first actions was to drive the 2004 race course the good, old-fashioned way, with a person behind the wheel. To collect data, we bolted four laser range finders to the roof, plus a GPS system for positioning. As it turned out, we never again looked at the data collected that day. But we learned many important lessons. The area where Sandstorm had faltered was really difficult, and even getting there had been a tremendous achievement for the CMU team. And the course was long: it took us about 7 hours do drive the entire 142 miles of the 2004 Grand Challenge course, which made the DARPA-imposed time limit of 10 hours feel uncomfortably tight. And we learned a first hardware lesson, one of many more to come. A few dozen miles into the course, one of our lasers almost fell off the roof; others came loose to the point that the data became unusable.

The remainder of the summer months were spent purchasing vehicle components and designing some of Stanley's hardware. The VW engineers quickly identified an appropriate version of the Touareg, flew it across the Atlantic, and began the development of the drive-by-wire interface. In designing Stanley, I strongly believed that the process of software development would be as important as the final "product." Hence, Stanley had to be designed to facilitate the development work, not just to survive the race. Taking the robot out for a spin should be as easy as driving a regular car. And it should be fun, so that we would do it often. Thus, in contrast to several of the robots at the 2004 race, which were unfit for human driving, Stanley remained street-legal, just like a regular car. By moving all computers into the trunk, we maximized the development space inside. During testing, a big red button became our life insurance when the robot was in control. This button enabled us to take over control at any time, even at high speeds. My hope was that at times where our robot software malfunctioned, the person behind the wheel would take over fast enough to avoid any damage.

CS294: Projects in Artificial Intelligence

As a first step towards building Stanley's software, we decided to throw together a quick end-to-end prototype robot (see figure 1). This robot was to be deficient, but it should contain all essential components of a race-capable vehicle. We drew our motivation to start with an integrated system—instead of spending our time on component technologies—from the fact that we were quite ignorant about the nature of this challenge. Only from running a robot through actual desert terrain, and by watching it fail, would we be able to tell where the real challenges lay. Cedric and Carlo fully supported this vision and rushed the development of a drive-by-wire system for the Touareg, making the robot car available in early October.

We now faced the question as to how to find the necessary manpower for building a first prototype. As a college professor, I decided to offer the project as a course. In this way, our small team could draw many new students into this project for a limited period of time. CS294, Projects in Artificial Intelligence, was offered as a graduate course in the fall quarter of 2004, which ran from October through December.

CS294 was not a common course. There was no textbook, no syllabus, no lecture. With the exception of the papers by Kelly and Stentz (1998a and 1998b) we didn't even read a paper, so that we wouldn't be biased towards a particular approach. Instead, the course was to be a team of students working together to build a robot in record time. Most students had never worked on robotics, and few had ever been part of a large team. So the course offered a totally new type of experience.

The course was open to all students on campus. From the nearly 40 students who showed up on the first day of class, 20 chose to stay on. We, that is Mike, me, and David Stavens, one of my Ph.D. students, ran this course. To manage such a large group of students, we divided the team into five groups, focusing on vehicle hardware, computing system, environment perception, motion planning, and low-level control. The first homework assignment was to have groups design their own work plan. Stu-

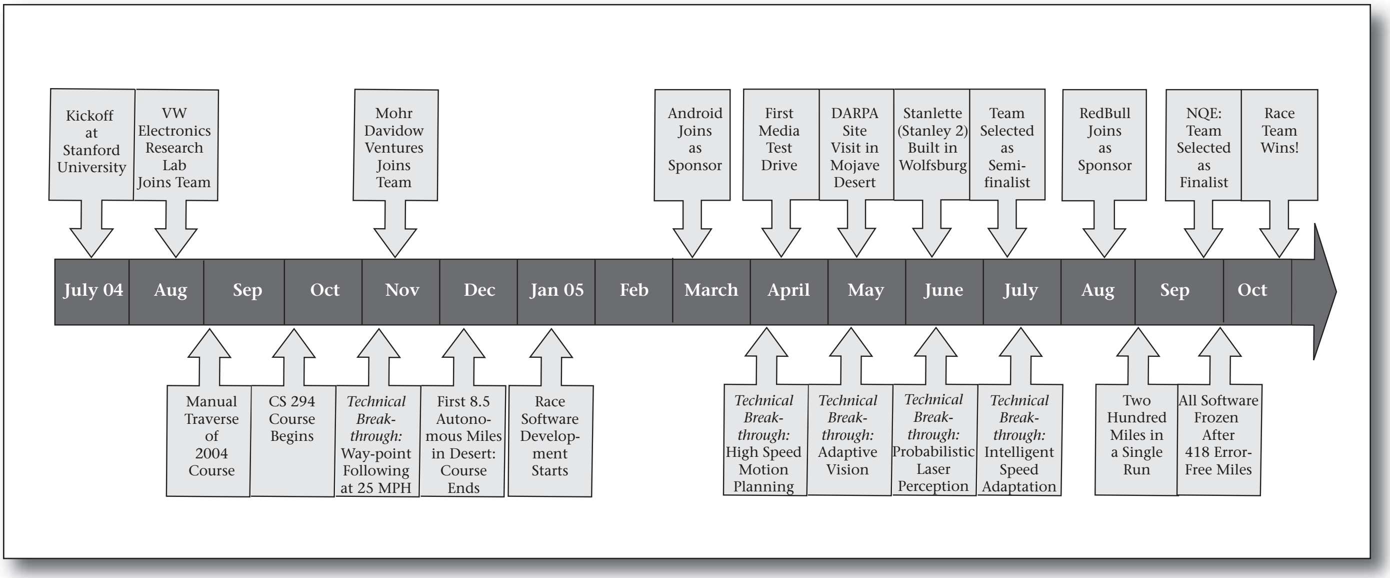

Figure 1. Approximate Timeline of the Stanford Racing Team.

dents had to come up with specifications of their contribution, a time line, and a sequence of milestones. This was a bit of culture shock for many students. Why didn't the instructor simply tell them what to do?

The initial two weeks of the course were used for groups to develop this work plan, and for negotiating interfaces with one another, so that all contributions would eventually work together. We jointly developed an interface document, along with a time line for the development process in the form of a Gantt chart. The class as a whole had two joint milestones: an initial event in which Stanley was to drive an autonomous mile on campus, and a final "autonomous desert mile" in the Mojave desert, scheduled for December 1, 2004.

For many students, milestones and Gantt charts were uncharted territory. In fact, few had ever worked in a large team. After all, defining milestones and setting time lines is entirely different from solving homework assignments. But teams quickly converged to an initial Gantt chart full of subtasks and minimilestones.

From the third week on, students focused on the technical work. In our class sessions, each group reported on progress, and we spent time resolving possible integration problems. In each session, I made a point of comparing each group's progress to our anticipated time line in the Gantt chart, so that we all could understand the rate of progress. In fact, the Gantt chart became my main method of spotting problems.

And problems were manifold. Over and over again, students proposed a technical idea, implemented it, and then observed it to fail in the field. Some students showed tremendous skills in identifying the underlying problems and solving them; others focused their energies on "explaining away" problems, or found reasons to conclude that the failure wasn't theirs. Conveying to students that as members of a team, it didn't matter who was at fault, required some efforts on my side. But in the end, most students got the message.

As the chief instructor, I worked hard to define and enforce deadlines and to keep the team focused. I also tried to establish a systematic testing methodology. Students had to define performance measures and use them as debugging tools. Measuring performance was entirely new to most students in class, and it was met with fierce resistance. It was as if I questioned the students' abilities to decide by themselves what works and what doesn't work. But as some of the better students noticed, this was the only way to spot bugs in the code and to understand the side effects of unsuspicious changes. I did not succeed in every instance in persuading the students to measure performance. But in those cases where I did, the result became noticeably stronger.

After many late evenings and weekends, the students managed to "deliver" the first integrated system just one week behind schedule. In mid-November, Stanley successfully drove its first mile, in many small loops around our garage. But many aspects of the system were still flawed. At times Stanley ignored obstacles, or simply drove off into random directions. So

Figure 2. The Final Day of CS294.

The class was getting ready for our first attempt to drive one autonomous desert mile. Stanley drove 8.5 miles that day, including some very harsh mountain terrain, but oddities in its control system limited its top speed to about 8 mph.

for the remainder of the course, the students worked mostly with us on improving the basic system components. The course meetings became obsolete, since the students preferred to spend their time in the shop. Each group would merely send the least important member as a “delegate” to class. Instead, most interactions took place in our garage, where students now worked around the clock.

Finally, on December 1, 2004, we all traveled to Barstow, CA, where the 2004 Grand Challenge had taken place. Figure 2 shows our group picture, taken at the starting line of the 2004 race. Stanley drove more like a drunken squirrel than a robot, but it was able to avoid obstacles and regulate its speed. At 8.5 miles into the course, we finally were forced to intervene when the car failed to decelerate on a steep mountain descent. Still, the sense of achievement was tremendous. Stanley had gone further than any other robot had gone in the Grand Challenge (albeit at a lower speed); and we had built an end-to-end system in just eight weeks!

Gearing Up

Following the course, we shrunk the team down to just a handful of the most trusted students, most notably Andrei Aron, Hendrik Dahlkamp, Philip Fong, Gabe Hoffmann, and Pascal Stang. Our criteria in admitting students to our team combined reliability, creativity, team play, and of course availability all the way into the race. Several senior researchers from Intel joined the team, led by Gary Bradski. Gary had developed a highly popular open source computer vision library[2] and was perfect to lead the computer vision team.

I now began to focus the team on the development of actual race software. Mike and I decided to toss most of the existing software, largely starting from scratch. The fact that we already had a first integrated robot made it obvious where work was needed. However, most of the existing software was not fit for the race.

From the beginning of this project, I believed that the weakest component of the system would be the one that determined the overall success of the mission. So rather than investing in the improvement of modules that already worked well, we focused our attention on the weakest module. Every time the robot drove off the road, or chose unreasonable speeds, we saw an opportunity for improvement.

Mike and I believed that quality control for software was of extreme importance. To this end, we created two software repositories, one for development software and one for race software. The development repository was for everyone to play with. The fact that even I had access to this repository became the content of a long-standing joke. For software to graduate into the race repository, it had to pass the scrutiny of a Mike Montemerlo. Mike, like no one else, truly understood what it took to make software reliable. Often he rewrote entire software packages from scratch, just so that the race software met his high standards. And he developed elaborate testing methods. So instead of all of us coding, Mike wrote most code, and the team's function was to make Mike as effective as possible. The only exception here was the computer vision software, which Mike never touched. But he made sure that this software could do no harm to the robot if it malfunctioned.

The period between January and May 2004—the date of the DARPA site visit—was a continual repetition of the following cycle: a brief trip to Barstow in the Mojave desert to test a new software revision and collect data, followed by an analysis of what went wrong, which then led to an extensive coding period back at Stanford. Each cycle took between two and four weeks, so we frequently embarked to Barstow, where the 2004 race had begun. The team held five weekly meetings, mostly for coordination. My role in these meetings was largely to focus the team on open problems, to set deadlines, and to enforce proper data-driven evaluation. Since our Grand Challenge team had gained a lot of visibility on campus, all sorts of individuals showed up for the team meetings. There was no shortage of creativity; in fact, there often was an abundance of creativity—possibly because we train our students to constantly come up with new ideas. Hence one of my many jobs became to shield the team from the too-many-ideas syndrome—a trap that could have easily delayed the development by many months. Mike and I had to focus all creative energy of the team into actual problem-solving activities, even in the early development phase.

In May, we faced our first serious hurdle: the site visit. DARPA had scheduled site visits with more than 100 teams, to select 40 semifinalists. Our site visit took place near the original race course in Barstow, and we passed with flying colors. Despite a broken alternator that required some last-minute rewiring, Stanley mastered the short course with perfection.

Making Race Software

After the successful DARPA site visit, we shifted our focus on finalizing the race software. From now on, most work took place in a desert. We initially used the 2004 Grand Challenge course near Barstow for testing and development; in early July Stanley drove the course end-to-end. At the end of July, however, DARPA banned all teams from the Mojave Desert, and we moved our operations to Arizona. On Carlo's initiative, VW had offered us shelter at its Arizona Proving Grounds. So the picturesque Sonoran Desert became our new playground, where we would stay in the months leading up to the race.

Interestingly, many of the key scientific innovations were made during this advanced development period that followed the site visit. And all of them were the direct results of actual problems encountered in extensive desert tests.

Problems were manifold. After spending months tailoring Stanley to drive in difficult mountainous terrain, we were stunned to see it fail catastrophically in the flats. The cause for this failure was an inability to detect berms in individual laser scans. Their specific shape made berms a subtle obstacle, much harder to

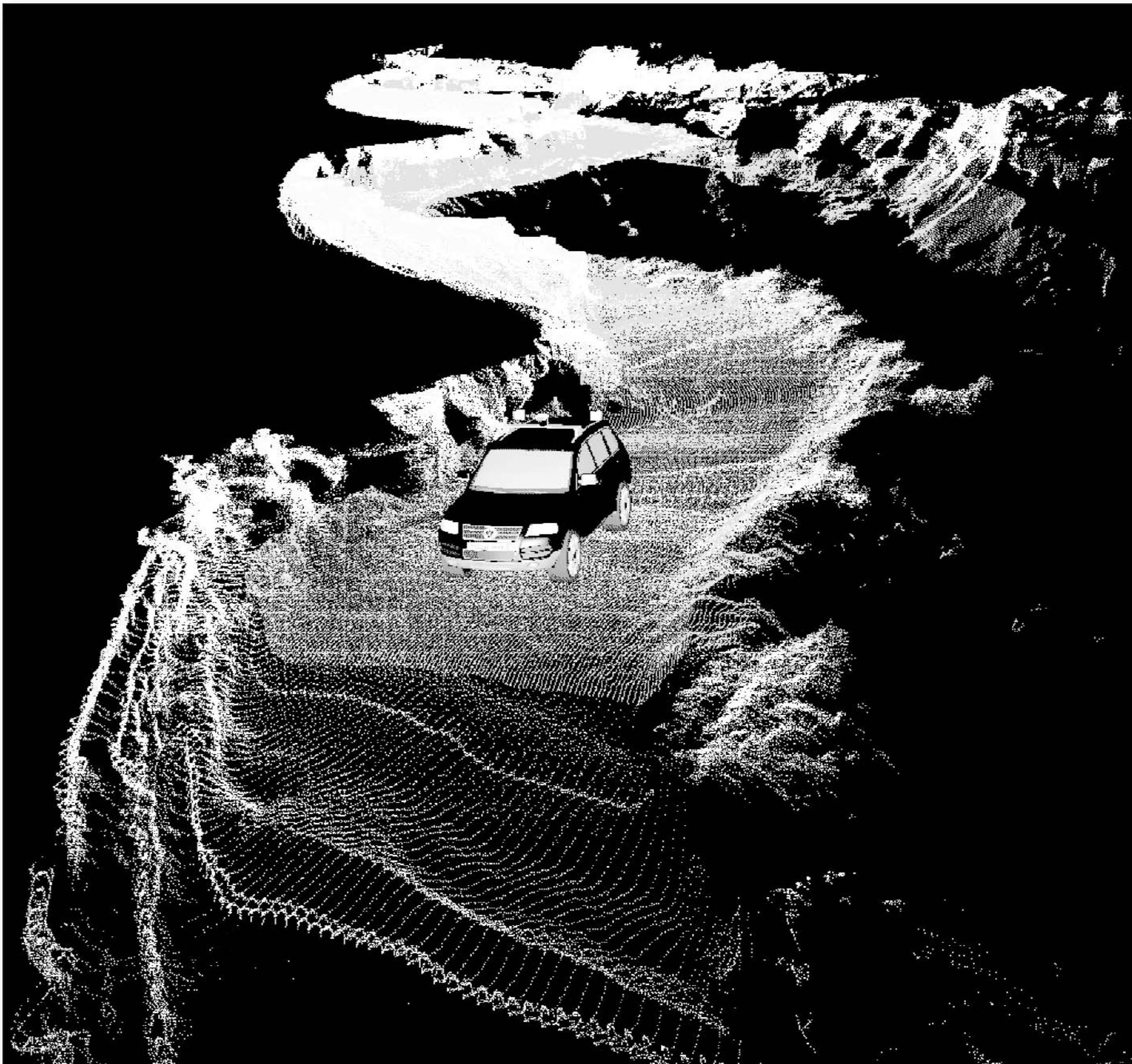

Figure 3. Laser Mapping.

As the robot navigates, its lasers acquire a point 3-D cloud of the terrain ahead. Stanley's main problem: its lasers see only 20 meters ahead, which is too short for fast driving.

detect than the rocks and plants we had encountered in the mountains. This problem eventually resulted in a complete overhaul of our mapping software. The new mapping software memorized laser scans to acquire full three-dimensional models of the road surface and applied advanced statistical tests for finding obstacles (Thrun et al. 2006), as illustrated in figure 3.

Equally surprising were the failure modes of our computer vision software. No matter how hard we tried, none of our road-detection algorithms could handle the many different road surfaces and the range of light conditions one could expect during the course of a day in the desert. Starting in January, the computer vision team began to search for new means to find roads on ever-varying lighting conditions. The decisive idea of adaptive vision (Dahlkamp et al. 2006) was due to Bob Davies from Intel. In adaptive vision, the camera module relied on the laser data to adapt its model of the road surface, as illustrated in figure 4. Thus, instead of solving the general computer vision problem, it solved the problem of "finding more of the same," but at a range beyond that of the laser range finders. The results were stunningly good: adaptive vision enabled Stanley to see obstacles at the 80 meter range. After an extensive empirical comparison, we settled on an implementation by Bob's colleague Adrian Kaehler. This decision was the result of an extensive empirical comparison, which I had asked Hendrik to pursue. In this comparison, Adrian's software beat Bob's in reliability and simplicity by a razor-thin margin.

We also learned that driving fast, right at the speed limit, could be a very bad idea. Sitting inside the vehicle while under computer control, and watching it running over ruts at high speeds like a dumb robot, convinced us that intelligent speed control was as essential for the health of the vehicle as steering control. This led David to invent a variable speed controller that slowed the vehicle down in response to environment features (Stavens and Thrun 2006). This invention was remarkable, in that none of us had even thought that speed was a research issue. All our work up to that point had focused on intelligent steering control, but speed proved equally important.

And finally we realized that the vast majority of obstacles were found at the side of the road, not at its center. By driving right on the road center, we could avoid nearly all obstacles without even having to take corrective action. Once we realized this, Mike developed a probabilistic road boundary filter that would track the principal boundary of the road over longer periods of time. He modified his planning routines so that the robot would always nudge back to the road center, even when an occasional obstacle forced Stanley to swerve to the road side. The day Stanley started actively seeking out the road center was a big day for us. This heuristic substantially improved the vehicle's success rate.

But not all moments were moments of fame and glory. We spent endless hours chasing embarrassing bugs, of the type that can never be reported in a scientific document. One was a bug in our power system, for which Stanford was responsible. The bug led the load on the alternator to oscillate between zero and 1000 watts, in about 1-second intervals. This discovery ended a sequence of mysterious alternator failures that had stunned the VW staff. Another bug pertained to the synchronization of clocks on different computers, which occasion-

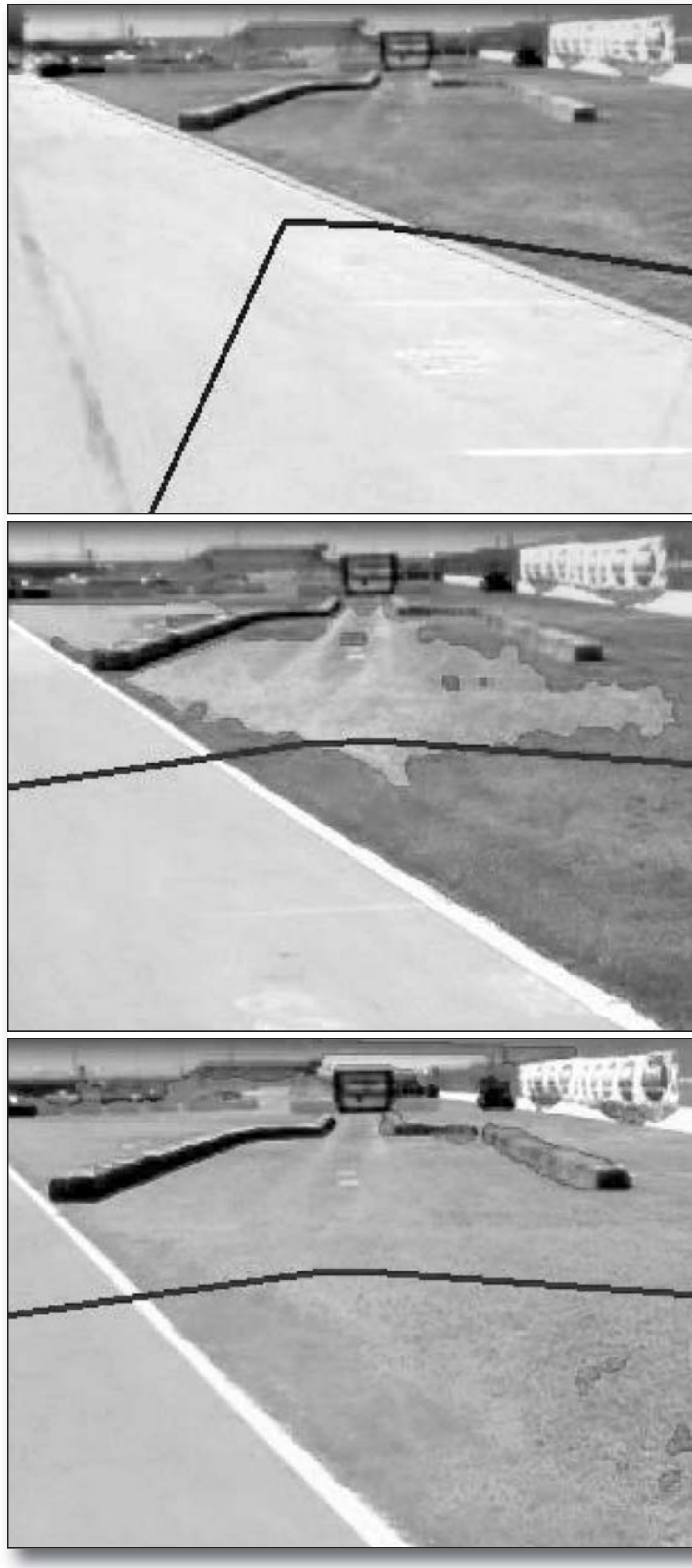

Figure 4. Adaptive Vision

The robot's internal model of drivable surface adapts as it goes. Shown here is a transition from pavement to grass. Training data is acquired using the short range laser model and, thanks to adaptive vision, the robot can find drivable terrain at a longer range ahead.

ally resulted in time going backwards by half a second. Since time stamping was an important aspect of our software, this bug had caused Stanley to see large phantom obstacles on the road, big enough to force the robot off the road. And we rediscovered the concept of ground loops. A ground loop involving the inertial measurement unit forced us to power-cycle this unit whenever we actuated the brake. Since most of us were software experts, we had been puzzled by the mysterious IMU crashes for many months, until one of us remembered the existence of the ground loop concept from a long-ago engineering class.

During this phase of the project, everyone on the core team fully understood what it meant to play with the team. Getting lunch for the team was as noble a deed as writing cutting-edge software. To the present day, I continue to be amazed by the willingness of every single team member to do whatever I asked him or her to do. And I tried to lead by example. My personal highlight was the day I spent building a tank trap out of PVC pipes. After bolting together three pipes, my team noted that the surface wasn't sufficiently similar to rusty metal. So I went back to the store to buy spray paint, and then spent hours applying a combination of paint and dirt to give the trap the look of a World War Two tank trap. This was not exactly the type job for which I had come to Stanford. But it was magically gratifying to keep my hands dirty and to spend my time on mundane things of no scientific value whatsoever. Vaughan Pratt, an esteemed Stanford colleague and dedicated team member, was equally willing to "waste" his time. Vaughan was always ready to drive a chase vehicle or to fix an engine problem—whatever was needed to promote the team.

As work progressed, Mike and I became the chief naysayers on the team. Some of the decisions we made were difficult. On several occasions, I had to confront a team member with the decision not to incorporate his software, on which he had worked for months. It's not easy to say "your software is great, but we will not use it on our vehicle, and we will all be better off for that." But I had to. Mike made similarly tough decisions, at times discouraging any of us (myself included) from going down tangential paths. I realize that many of our decisions came as a blow in the face of some of our very best team members. But Mike and I always acted in the belief that the disappointment of excluding someone's work was smaller than the disappointment of losing the race for failing to make such decisions. And the team seemed to understand.

My other key role was bug discovery. Many of us spent endless hours analyzing data logs, and so did I. Mike had written a suite of data logging and visualization tools, and we always logged various data streams when in autonomous mode. When a surprise happened, I spent endless hours trying to explain the event by analyzing the data logs.

Most bugs were easy to find. But some, like the time-goes-backwards bug, plagued us for months. This bug was discovered only by coincidence, when I analyzed the unrelated IMU ground-loop problem. The day we found this bug was another very big day for the team. Even though no paper can be written about the endless hours of analyzing data logs, I view data analysis as a prime intellectual activity and to the present day take great pride in the bugs I found.

Fund-Raising and Media Events

Parallel to the technical developments, one of my jobs was to oversee the corporate and media relationships of the team. Because the use of federal money was not legitimate, the entire development had to be funded through other means. We had to identify a number of sponsors for the team. The early cooperation with VW was a tremendous boost for the project. Still, we needed money for salaries, equipment, and travel.

I decided not to ask Stanford for money, as Stanford had just supplied me with startup funds. Instead, I told my dean's staff that I'd use my remaining startup funds, joking that I'd simply join a different university for new startup money once those funds were depleted. The dean eagerly offered me support of his professional fund-raising staff, led by Laura Breyfogle. Within days, her team identified two new main sponsors, David Cheriton, a colleague who was also involved on the technical side of things, and Mohr Davidow Ventures (MDV). MDV had a history of sponsoring races: it won three legendary "Sand Hill Road Races," which were local races involving gravity-propelled vehicles. An early fourth sponsor was Andy Rubin, a long-time friend and robot enthusiast who had just profitably sold his startup company Android. For Andy, the "condition" of sponsorship was for us to win the Grand Challenge. Simple as that.

MDV advised us from the beginning to put in place a written document detailing the obligations and rights of individual sponsors. In this document, we granted sponsors visibility on the vehicle and all promotional materials, the permission to use Stanley in their own pro-

Figure 5. Some of the Core Team Members at the Final Development Day in the Sonoran Desert.

Rear: Mike, Joe, and David. *Front:* Hendrik and me; Sven is absent. As all code is now frozen, most team members will return home to take time off before the race.

motional activities, plus a veto right to block new sponsors when conflicts arose. There were no deliverables or statements of work in this agreement, and all intellectual property would remain entirely with Stanford.

In addition to cash donations, we also received a number of equipment offers from high-tech companies eager to place their products on our vehicle. This was a tricky issue. Naturally, there exists a myriad of technologies that are almost right for the Grand Challenge, but not quite. Mike, in his infinite wisdom, cautioned me against accepting such gifts. Jokingly he said that one way to slow down competitors was to give them lots of free equipment, so that they would spends all their time integrating. And right he was. Deviating from this rule, we once decided to accept a state-of-the-art stereo vision sensor. After several weeks of hard work we decided not to use it, since it added too little to the laser range finders. This was a bit of a setback to the sponsor and to the students who had spent weeks on the integration. Ever since this experience, cash donations became the only acceptable form of sponsorship.

The fund-raising activities continued all the way up to the race. The two last companies joining the circle of corporate sponsors were RedBull and Intel. Because both of them signed up very late in the game, the vehicle liverage was only determined a few weeks before the race. In record speed, VW engineer Sven Strohband, who had taken over as lead vehicle engineer from Cedric Dupont, designed and implemented the final liverage.

The corporate sponsors played multiple important roles in this project. Apart from various technical and financial contributions, they were essential for shaping the media relationship of the team. Pamela Mahoney from MDV took the lead of the communications team, but VW and Stanford were also actively involved. Just before the DARPA site visit, the communications team launched a first media briefing, which took place on Stanford's campus. As the race drew closer, Pam effectively become a full-time team member, and handled our media relations with great skill.

At some point VW flew in media experts from their headquarters in Auburn Hills,

Figure 6. The Team at the Beginning of the National Qualifying Event held at the California Speedway, Fontana California.

Since the system was completely developed at this time, the team had little to do.

Michigan, to prepare the Stanford Racing Team for the upcoming media interactions. As engineers, we had little if any prior exposure to journalists. Several of us gave mock interviews that were videotaped and critiqued by VW's communications experts. And indeed, the training taught us a great number of lessons. It helped us in conveying an upbeat message and one that would be consistent for the entire team. Because AI was such an essential component of the Stanley robot, most of our technical briefings focused on the importance of AI in the robot software.

Testing, Testing, Testing

One of the great decisions the team had made early on was the creation of a testing group. The task of the testing group was to scrutinize the work of the developer team through extensive testing. This type of testing was quite complimentary to the type tests conducted by the development team. The purpose of the testing group was to poke holes into our approach and to catch things we had overlooked in the development. To this end, the individuals in the testing group were intentionally not part of the software development team, providing them only with minimal knowledge about the actual workings of the system. Our hope was that the testing team would be as unbiased as possible when setting up the tests.

The testing group's first contribution came just before the site visit. At a point where we, the developers, were quite confident of Stanley's fitness for the site visit, the testing group organized a mock site visit, with Celia Oakley, a Stanford alum, assuming the role of the DARPA program manager. Needless to say, the event was a disaster. Bits and pieces of the software malfunctioned or were not in compliance with DARPA's site visit requirements. We repeated the drill a number of times, until we finally satisfied our testing group. By the time the real site visit occurred, we were confident that Stanley would run like a charm. And so it did.

For the actual race, the testing group had prepared a written testing document. The document meticulously spelled out tests on all aspects of the robot, on more than 150 pages! Some tests looked silly to us, the developers, such as crossing a paved railroad track. Yeah, we knew Stanley could do that. Others were ingenious, in that they uncovered serious hidden bugs. It was amazing to see how many of the tests Stanley failed at first try. By the time the National Qualifying Event came along, we had fully executed the entire test document, and run every single test.

The testing group activities were complemented by our own experiments. In the final months, the development team spent most of its time in and around the car. Some of us used

Stanley's backup vehicle, dubbed "Stanlette," to acquire new test courses. Others ran Stanley through the desert or spent the day in Volkswagen's facility improving specific aspects of the system. The core team at this point was quite small: David, Hendrik, Mike, me, Sven, and Joe von Niekerk, an ingenious VW engineer (see figure 5). Other team members were routinely flown to Arizona as needed, sometimes on just a few hours notice. In the final weeks, Volkswagen flew in Lars Jendrossek, a Touareg expert from Germany. Lars helped fix some essential last-minute vehicle problems. And Carlo, the ERL director, spend extensive time in Arizona helping the team as well. We logged hundreds of miles in remote deserts near the Mexican border, always wearing clunky white helmets as a last means of protection against software bugs.

In these final months, the development team effectively did only two things: fixing major bugs and tuning critical parameters. A bug was major if it threatened to derail Stanley's bid for finishing this race.

Major bugs were easy to spot: At times Stanley would drive off the road or simply stop in the middle of a run. One day in September, we embarked on a day-long 150-mile trek around a dirt oval on Volkswagon's facility, just to learn that 141 miles into the course, a combination of a GPS shift and a software bug made Stanley stop, unwilling to forge ahead. The irony here was that the 2004 race was 142 miles long. With this bug, Stanley would have stopped just one mile away from the finishing line in the 2004 race.

On another day, a software bug resulted in Stanley accelerating down a narrow steep slope, instead of putting the foot on the brake—and as a result it crashed into a tree (unfortunately this incident happened the day two *New York Times* staff visited us to report on our progress and the challenge). And on yet another day Stanley encountered large puddles on the course, giving us an opportunity to test autonomous driving through water.

For parameter tuning, we spent endless hours measuring. We meticulously measured the speed at which Stanley could safely swerve around a frontal obstacles when traveling on gravel. Would it be 25 mph or 30 mph? And how much faster could we go if we used the adaptive vision module as an early warning system? Any such question required many hours of testing. When setting a parameter, we always tried to err in the direction of safety, so that Stanley would finish the race.

At the end of the testing ordeal, Stanley had logged hundreds of miles without any failure. Our confidence rose that the robot could actually finish the race. To quantify different terrain types, we had developed our own terrain taxonomy, with Grade One being wide-open flats and Grade Five being serious off-road terrain (the actual race only went to Grade Three). Stanley was able to drive confidently on Grade Four terrain, and even mastered patches of Grade Five. Stanley was never an elegant driver, in that it often swerved around obstacles much later than a human driver. This was particularly obvious in Grade One terrain, where Stanley drove fast and hence suffered from its relatively short perceptual range. But Stanley had clearly become a competent driver, safely avoiding collisions for hundreds of miles on end. And it did so in terrain that was significantly more difficult than that of the actual Grand Challenge course.

Mike and I believed that freezing the software was more important than making things perfect. In fact, we intentionally did not fix a number of known bugs and suboptimal settings. Instead, we focused on hardening the existing system, only changing parameters when absolutely necessary. More than a week ahead of the National Qualifying Event, we finished all testing and suspended all development activities. Stanley was done. In the few days remaining, it would have been impossible to recover from a potential accident. Hence the risk of further experiments outweighed their benefits. This left us a week with absolutely nothing to do.

The Race

The race event begun in September, with the National Qualifying Event (NQE) at the California Speedway; see figure 6. We arrived well rested. Most team members, myself included, had used the opportunity to return home to relax in the final week leading up to the NQE.

The time at the NQE was great fun. We found ourselves surrounded by hundreds of individuals who shared the same vision: making cars drive themselves. The competition was manifold. It ranged from well-staffed research universities to small teams of individuals, who at times made up a lack of robotic expertise with immense enthusiasm. I was particularly intrigued by two teams: Team Blue who had chosen a motorcycle as its race vehicle, and the Palos Verdes High School Team, composed mainly of high school students. Because we had little to do at the NQE, we spent a lot of time talking to other teams. Mike helped some teams debug last-minute laser problems. And Lars, our mechanic, spent more time with oth-

Figure 7. The Dreams Come True.

Stanley becomes the first robot to finish the DARPA Grand Challenge, winning by a razor-thin margin ahead of three other robots. It is a moment I will never forget.

er teams' vehicles than with Stanley—just so that he wouldn't get bored.

And boring we were, at least according to the crew that DARPA had recruited to clean up after each vehicle. Stanley was the only vehicle never to touch any obstacle in the NQE runs. Consequently, the cleanup crew called us "Team Boring." In the last two runs of the NQE, the course was identical. Stanley's finishing time in these two runs was also identical, up to the second. As if Stanley had achieved German precision.

However, Stanley was not the fastest vehicle. As a result, Stanley got the second pole position in the race, trailing CMU's H1ghlander and ahead of CMU's Sandstorm robot. After some internal discussions, the team resisted the temptation to change Stanley's parameters to make it go faster. Instead, we wanted to play it safe, maximizing Stanley's chances of finishing the race. As Mike pointed out, no robot had ever finished such a race before. So there was no point in taking on additional risks by going faster than previously rehearsed.

To me, the race was a thrill. In the first hours, I paid little attention to the vehicle's progress, barely noticing that H1ghlander pulled ahead of Stanley at a rate of a few minutes per hour. I spent most of my time inside a tent put up by DARPA, talking to journalists and some Silicon

Valley celebrities who had come to watch the race. At some point I even thought Stanley was out of the race. When consulting the display four hours after the start, I noticed Stanley had not moved for several minutes. Because we programmed Stanley to always move, my only explanation was that Stanley had encountered a terminal problem. Little did I know that DARPA had paused Stanley to give more leeway to CMU's H1ghlander robot.

And then the surprise happened. Teams that were faster than Stanley in the beginning of the race, such as Team Ensco's Dexter and CMU's H1ghlander, ran into problems. Dexter lost its ability to avoid collisions, and H1ghlander suffered an engine problem. At 102 miles into the race, Stanley managed to pass H1ghlander. This made Stanley the surprise front-runner. From this point on, our eyes were glued to the computer screens. As Stanley transcended a treacherous mountain road called Beer Bottle Pass, we could see from the live video feed that Stanley's sensors were slightly miscalibrated, making it hug the right side of the road just a little more than normal. But our navigation software caught these problems, and Stanley descended through this most challenging of all passes with its usual precision and agility. I had become so familiar with Stanley's habits that I immediately recognized the calibration problem. And I anticipated every one of Stanley's responses, at least so I believed.

An hour later Stanley returned to Primm, Nevada, where DARPA had placed the finishing line. The moment was magic: nearly seven hours after leaving the starting gate, Stanley had successfully navigated 131.6 miles of unrehearsed and, at times, punishing desert terrain. When I finally spotted Stanley only miles before crossing the finishing line, I couldn't fathom it. What looked impossible just a year ago had been achieved. As Stanley arrived, images like the one in figure 7 quickly went around the globe. From the five robots that finished, Stanley emerged as the fastest vehicle and won the race. It won the race just 11 minutes ahead of the second-fastest robot. Seeing Stanley return was one of the biggest moments of my professional life. And I am sure the same is true for my team, which had worked so hard for this final day.

We all realized that we were just incredibly lucky to come in first. Not even DARPA had expected such a tight outcome, and several other teams could justifiably claim that with minor changes of software or hardware their robots would surely have beaten Stanley. A postrace data analysis revealed that Stanley, too, suffered problems during the race that could have easily terminated its run very early on. In one encounter, Stanley actually drove on the berm for about a second—but luckily without any damage to the vehicle. As we now know from an extensive analysis of the log files, this specific problem had never occurred before. It was related to a timing problem of the incoming sensor data stream. Luckily, this problem only affected Stanley in the flats, where driving off the road caused no damage.

Because of this razor-thin victory, we quickly realized that the true achievement of the day had been the fact that five teams finished. This was a historical victory for an entire community of researchers, not just our team. What many had believed to be impossible had actually been achieved. And the fact that five robots finished underscored the progress the field had made in just a year. As Joe Seamans and Jason Spingarn-Koff from NOVA put it, "[s]ome day, the great robot race may well be remembered as a moment when the partnership between humans and the machines we create was forever changed."

What's Next?

I am now driven by the vision that one day, self-driving cars will change society (Dickmanns 2002; Hebert, Thorpe, and Stentz 1997; Pomerleau 1993). In the United States, some 42,000 people die every year in traffic accidents, mostly because of human error (the worldwide annual death toll is about 1 million people). A self-driving car could prevent many accidents and save thousands of lives. There are other benefits. Commuting takes up a significant fraction of time for many working people: by making cars drive themselves commuters could use their time in more productive ways. Self-driving cars would also allow people to get around who currently cannot drive, such as blind people, elders, and even children. Robotic precision on highways might increase the packing density and throughput of our nation's highway system and relieve the United States of one of its most critical infrastructure problems. And finally, a robotic car might park itself or drive itself to people in need of transportation.

In public presentations I often encounter skeptics and disbelievers. What about the fun of driving? Shall I really trust my life in the hands (or, better, wheels) of a robot? Skepticism is important, but having sat in a self-driving car for months on end, I have become a believer. There is something really powerful in the idea that every car comes with a chauffeur, which can be engaged at the driver's leisure. I believe this technology will some day change society, and we will all be better off for that.

At Stanford, we are poised to forge on. One of the critical limitations of the DARPA Grand Challenge was the absence of other traffic. It also didn't deal with the complexity of driving in cities. DARPA recently announced a new challenge that addresses these deficiencies. If successful, the Urban Challenge will push technology over a critical hurdle. And it will provide ample opportunity for basic scientific research in robotics and AI.

When a journalist called me up recently to ask if we'd be part of the Urban Challenge, I hadn't yet heard of the newly announced race. But I didn't think twice about how to respond. And now I find myself back at the very beginning of all of this, building up once again a new team of developers and sponsors for the next great robot race.

Acknowledgments

I am extremely grateful to the members of the Stanford Racing Team who made it all happen, and the various individuals who advised us along the way. In alphabetical order, I thank Alex Aiken, Philippe Alessandrini, Andrei Aron, Joe Bajurin, Joseph Baker-Malone, Daniel Barreto, David Black-Schaffer, Gary Bradski, Laura Breyfogle, Florian Buron, William Chan, David Cheriton, Jason Chuang, Hendrik Dahlkamp, Bob Davies, Miles Davis, James Diebel, Tom Dienstbier, Cedric Dupont, Scott Ettinger, Philip Fong, Paul Fontes, Jeanne Friedman, John Gale, Chris Gerdes, Morgan Halpenny, Susan Hansen, Gabe Hoffmann, Lars-Erik Jendrossek, Eric Jensen, Holly Jin, Adrian Kaehler, Christian Koelen, Joerg Kreft, Filip Krsmanovic, Jackie Lai, Kenny Lau, Jonathan Lee, Monica Mackey, Walter Macklem, Pamela Mahoney, Aditya Mandayam, Charles Markey, Michael Montemerlo, Ara Nefian, Andrew Ng, Joe van Niekerk, Celia Oakley, David Orenstein, Archan Padmanabhan, Mark Palatucci, Jim Plummer, Vaughan Pratt, Allen Rabinovich, Michael Saunders, Frank Scioscia, Curtis Spencer, Pascal Stang, David Stavens, Sven Strohband, Jason Swager, Maybelle Tan, Wei Wei, Chou-han Yang, Fiona Yeung, and Chih-Han Yu.

I also gratefully acknowledge the sponsorship by Volkswagen of America's Electronics Research Lab, Mohr Davidow Ventures, Android, Red Bull, Intel Research, Honeywell, Tyzx, and Coverity. Generous financial contributions were also made by David Cheriton, the Johnson Family, and Vint Cerf. Finally, I am deeply indebted to DARPA for organizing this event. Without DARPA's vision and determination, this race would not have been possible. Numerous government staff deserve credit, including Tony Tether, Ron Kurjanowicz, and Norm Whitaker.

Note

1. www.darpa.mil/grandchallenge05/Rules8oct04.pdf.

2. See the Intel Open Source Computer Vision Library, by G. R. Bradski. www.intel.com/technology/computing/opencv/index.htm.

References

Dahlkamp, H.; Kaehler, A.; Stavens, D.; Thrun, S.; and Bradski, G. 2006. Self-Supervised Monocular Road Detection in Desert Terrain. Paper presented at the Robotics Science and Systems Conference, 16–19 August, Philadelphia, PA.

Defense Advanced Research Projects Agency. 2004. *DARPA Grand Challenge Rulebook, 2004*. Washington, DC: U.S. Government Printing Office.

Dickmanns, E. D. 2002. Vision for Ground Vehicles: History and Prospects. *International Journal of Vehicle Autonomous Systems,* 1(1): 1–44.

Hebert, M.; Thorpe, C.; and Stentz, A. 1997. *Intelligent Unmanned Ground Vehicles: Autonomous Navigation Research at Carnegie Mellon University.* Dortrecht, The Netherlands: Kluwer Academic Publishers.

Kelly, A., and Stentz, A. 1998a. Rough Terrain Autonomous Mobility, Part 1: A Theoretical Analysis of Requirements. *Autonomous Robots* 5(2): 129–161.

Kelly, A., and Stentz, A. 1998b. Rough Terrain Autonomous Mobility, Part 2: An Active Vision, Predictive Control Approach. *Autonomous Robots* 5(2): 163–198.

Montemerlo, M.; Thrun, S.; Dahlkamp, H.; Stavens, D.; and Strohband, S. 2006. Winning the DARPA Grand Challenge with an AI Robot. In P*roceedings of the Twenty-First National Conference on Artificial Intelligence.* Menlo Park, CA: AAAI Press.

Pomerleau, D. 1993. *Neural Network Perception for Mobile Robot Guidance*. Boston: Kluwer Academic Publishers.

Stavens, D., and Thrun, S. 2006. A Self-Supervised Terrain Roughness Estimator for Off-Road Autonomous Driving. Paper presented at the Conference on Uncertainty in AI (UAI), Cambridge, MA, 13–16 July.

Thrun, S.; Montemerlo, M.; and Aron, A. 2006. Probabilistic Terrain Analysis for High-Speed Desert Driving. In *Proceedings of the Robotics Science and Systems Conference,* ed. G. Sukhatme, S. Schaal, W. Burgard, and D. Fox. Cambridge, MA: The MIT Press.

Thrun, S.; Montemerlo, M.; Dahlkamp, H.; Stavens, D.; Aron, A.; Diebel, J.; Fong, P.; Gale, J.; Halpenny, M.; Hoffmann, G.; Lau, K.; Oakley, C.; Palatucci, M.; Pratt, V.; Stang, P.; Strohband, S.; Dupont, C.; Jendrossek, L.-E.; Koelen, C.; Markey, C.; Rummel, C.; van Niekerk, J.; Jensen, E.; Alessandrini, P.; Bradski, G.; Davies, B.; Ettinger, S.; Kaehler, A.; Nefian, A.; and Mahoney, P. 2006. Stanley, the Robot That Won the DARPA Grand Challenge. *Journal of Field Robotics*.

Urmson, C.; Anhalt, J.; Clark, M.; Galatali, T.; Gonzalez, J. P.; Gowdy, J.; Gutierrez, A.; Harbaugh, S.; Johnson-Roberson, M.; Kato, H.; Koon, P. L.; Peterson, K.; Smith, B. K.; Spiker, S.; Tryzelaar, E.; and Whittaker, W. L. 2004. High Speed Navigation of Unrehearsed Terrain: Red Team Technology for the Grand Challenge 2004. Technical Report CMU-RI-TR-04-37, Robotics Institute, Carnegie Mellon University, Pittsburgh, PA.

Sebastian Thrun (thrun@stanford.edu) is an associate professor of computer science and electrical engineering at Stanford University, where he also serves as the director of the Stanford AI Lab. His research focuses on robotics and AI.