Self-supervised Monocular Road Detection in Desert Terrain

Hendrik Dahlkamp*, Adrian Kaehler†, David Stavens*, Sebastian Thrun*, and Gary Bradski†
*Stanford University, Stanford, CA 94305
†Intel Corporation, Santa Clara, CA 95052

***Abstract*— We present a method for identifying drivable surfaces in difficult unpaved and offroad terrain conditions as encountered in the DARPA Grand Challenge robot race. Instead of relying on a static, pre-computed road appearance model, this method adjusts its model to changing environments. It achieves robustness by combining sensor information from a laser range finder, a pose estimation system and a color camera. Using the first two modalities, the system first identifies a nearby patch of drivable surface. Computer Vision then takes this patch and uses it to construct appearance models to find drivable surface outward into the far range. This information is put into a drivability map for the vehicle path planner. In addition to evaluating the method's performance using a scoring framework run on real-world data, the system was entered, and won, the 2005 DARPA Grand Challenge. Post-race log-file analysis proved that without the Computer Vision algorithm, the vehicle would not have driven fast enough to win.**

I. Introduction

This paper describes a Computer Vision algorithm developed for our entry to the 2005 DARPA Grand Challenge. Our vehicle (shown in Fig. 1) uses laser range finders as sensors for obstacle avoidance. Their range, however, is very limited due to high levels of environmental light, a shallow incident angle with the ground and possible dirt on the sensors. Since a vehicle needs to be able to stop within this distance, using lasers limits the maximum speed at which a vehicle can travel while maintaining the ability to avoid obstacles.

Therefore, we are proposing a camera system as an additional long-range sensor. Our goal is to detect drivable surface in desert terrain, at long range. In the existing body of literature, the primary concern has been the identification of the kind of paved roads observed in daily situations such as highways, country roads and urban roads. Such roads are typically characterized by their asphalt or concrete color, as well as by standardized markings in or at the edges of the roads, and are hence much easier to identify than undeveloped dirt-roads occurring in the desert area where the Grand Challenge took place. Consequently, most published papers are not directly applicable to the problem:

Universität der Bundeswehr München (UBM) has one of the most mature autonomous driving systems, in development since 1985[1]. Their approach involves modeling roads as clothoids and estimating their parameters using an Extended Kalman Filter (EKF) on visual observations and odometry. More recent work extends that framework to dirt road detection[2], waypoint driving[3], and obstacle detection while driving off-road[4].

Fig. 1. Image of our vehicle while driving in the 2005 DARPA Grand Challenge.

For the DARPA Grand Challenge, however, their system[5] is not directly applicable: Their road finder operates on grey, not color images. The desert terrain relevant for the Grand Challenge has very subtle color tones hidden in grey images. Furthermore, success of dirt-road finding depends on a manual initialization step. Finally, the system cannot handle arbitrarily shaped surfaces, even the dirt-road localization needs a road complying to model assumptions.

Both the *ARGO* project of the Università di Parma, Italy[6], [7], as well as Carnegie Mellon University (CMU)'s *NavLab* project[8] are only concerned with on-road driving. In addition to using tracking of lane markings, CMU also deals with inner-city driving involving cluttered scenes and collision avoidance[9].

Another system applicable to inner-city driving belongs to Fraunhofer IITB in Germany. Their use of computer vision for road finding includes edge extraction and fitting those to edges to a detailed lane- and intersection model for pose estimation[10].

The DARPA Learning Applied to Ground Robotics (LAGR) project has also created interest in vision-based terrain classification[11]. Due to the relatively slow speed of the LAGR vehicle, however, sensor lookahead distance in the LAGR project is not as crucial as in the DARPA Grand Challenge. Because LAGR sensors are limited to stereo cameras and a contact sensor, combination of different sensor modalities is not possible.

Outside of robotics, a surprisingly similar application is skin-color detection. Previous work, e.g.[12] also uses a gaussian-mixture model to deal with varying illumination

(though not desert sun) and varying skin tone between individuals, comparable to road material changes. There, however, the option of fusing data with a different sensor modality does not exist.

Finally, previous work associated with our group[13] provides very good classification in the desired terrain types, but needs examples of non-drivable terrain as training data, which our robot cannot easily provide automatically.

To overcome the limitations of these published algorithms, and achieve the level of robustness and accuracy required for the DARPA Grand Challenge, we have developed a novel algorithm whose main feature is the adaptation to different terrain types while driving, based on a self-supervised learning algorithm. This self-supervised learning is achieved by combining the camera image with a laser range finder as a second sensor. Specifically, the laser is used to scan for flat, drivable surface area in the near vicinity of the vehicle. Once identified, this area is assumed to be road and used as training data for the computer vision algorithm. The vision algorithm then classifies the entire field of view of the camera, and extracts a drivability map with a range of up to 70m. The combined sensors are integrated into a driving strategy, which allows for very robust, long range sensing. This is ultimately proven by winning the 2005 DARPA Grand Challenge.

Section II of this paper explains the different steps of this algorithm in detail, Section III provides numerical results that lead to parameter selections for the race and Section IV presents results achieved by running the algorithm in the 2005 DARPA Grand Challenge.

II. Algorithm Description

The overall method consists of seven major parts. These are: (A) Extracting close range road location from sensors invariant to lightning conditions, (B) Removing sky and shadow areas from the visual field, (C) Learning a visual model of the nearby road, (D) Scoring the visual field by that model, (E) Selecting identified road patches, (F) Constructing a sky-view drivability map, and (G) exploiting that map for a driving strategy. We will address each of these separately.

A. Extracting close range road location from laser sensors

Scanning laser range finders mounted rigidly on a moving vehicle are used to sweep out a map of the terrain in front of the vehicle one line at a time. Referencing the laser scans with an exact 6 degree of freedom (DOF) pose estimate, the accumulation of such lines form a point cloud corresponding to the locations of individual points on the surface of the ground, or obstacles ahead. Using this point cloud, a 2-dimensional drivability map is established, dividing the area into drivable and non-drivable map cells. Our approach, described in [14], achieves this by looking for height differences within and across map cells while modeling point uncertainties in a temporal Markov chain. Once such a map is created, a quadrangle is fit to the largest drivable region in front of the vehicle. This quadrangle, usually shaped like a trapezoid, is shown in black on the left side of Figure 2.

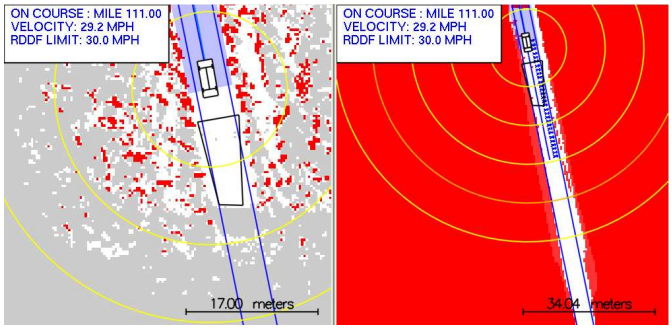

Fig. 2. Real-time generated map of the vehicle vicinity.
The left image shows close-range map as generated by the laser scanners. Red cells are occupied by obstacles, white cells are drivable and grey cells are (as yet) unknown. The black quadrangle is fit into the known empty area and shipped to the computer vision algorithm as training data for drivable road surface.
The right image shows a long-range map as generated by the computer-vision algorithm described in this paper. It can be seen that the road detection range is in this case about 70m, as opposed to only 22m using lasers.

Using the 6 DOF position estimate for the vehicle and the position of the camera with respect to the vehicle, we project the quadrangle into the visual plane of the camera. It is now safe to assume that this area in the image contains only drivable surface or road, because lasers just identified it to be flat and because GPS data indicates that we are inside the Grand Challenge driving corridor.

We will hence forth refer to this quadrangle area as the ”training” area: based on this area we will build our models of what the road ”looks like”.

B. Removing sky and shadows

We must deal with cast shadows that are dark enough that anything in them is hidden. In our present case, we opt to simply remove shadowed terrain. This can cause occasional problems, in particular when a long shadow falls across a road, but for our present purpose the option of eliminating shadow pixels from the model provides acceptable results.

We define a shadow pixel as any pixel whose brightness falls below a minimum threshold, and whose blue content is greater than its red or green content (measured as 8 bits in each of the three channels). The sky is not problematic for the algorithm, but measurement of sky pixels against the learned models (see below) is computationally wasteful. We implement the horizon finding algorithm originally proposed by Ettinger et al.[15] to eliminate all pixels above that horizon. We also use a flood fill to find pixels below the horizon which are the same color as the sky above, and also eliminate these. Finally, we found that a dilation of the sky thus defined removes pixels on distant horizon objects, saving compute time at no loss to the algorithm otherwise. The right part of Fig. 4 shows a sample sky-ground segmentation.

C. Learning a visual model of the nearby road

Once we have the training area, and have removed any shadows from that training area, the next task is to abstract

the data into a model which we subsequently use to score pixels outside of the training area. Our basic model of the road appearance is a mixture of Gaussians (MOG)-model in RGB space, with k Gaussians to be found. Our method is to classify all pixels in the training area using k-means clustering[16], and then to model each cluster by its average value, its covariance matrix, and its mass (where the mass is defined as the total number of pixels in the cluster). These abstractions are the training models. In addition to the k training models, the method provides for 'n' learned color models (where $n > k$). Initially, all of these learned models are null. Each of the training models is compared to the learned models at any given frame of the incoming video stream. If the training models overlap with a learned model according to the following relation:

$$(\mu_L - \mu_T)^T(\Sigma_L + \Sigma_T)^{-1}(\mu_L - \mu_T) \leq 1 \quad (1)$$

(where μ_i and Σ_i are the mean and covariances of the two models respectively) the training model is interpreted as new data for that learned model, and the learned model is updated. The model is updated according to the formula

$$\mu_L \leftarrow (m_L\mu_L + m_T\mu_T)/(m_L + m_T) \quad (2)$$
$$\Sigma_L \leftarrow (m_L\Sigma_L + m_T\Sigma_T)/(m_L + m_T) \quad (3)$$
$$m_L \leftarrow m_L + m_T, \quad (4)$$

where m_i is the mass of the associated model. We acknowledge there are statistically more sound ways of integrating these. If the training model matches no existing learned model, then one of two things happens: if there are still null models one of them is replaced by the training model. If no null models remain, then the learned model with the lowest mass is eliminated and replaced with the training model. In this way, new road types which are encountered generate new models until the space for storing models is saturated. Once this space is saturated, new models can still be learned, but known models which are not associated with a large mass (i.e. a large amount of experience of that model) will be lost to make room for the new model.

The setup of distinguishing k gaussians describing the current training area from n gaussians describing training history allows the system to tightly fit its model to the training while keeping a richer history of the past road appearance. In addition, the matching process allows for dynamic adaptation to situations of both slow-varying and quickly changing road appearance. In the first case of smooth trail driving, the training model is matched to a learned one and updates it slowly by merging data. In the second case of a sudden road surface change, the system reacts instantly by replacing an old learned gaussian with a new one from the training area.

D. Scoring the visual field by the road model

At this point, it is possible to score all pixels in the image according to the learned models. Individual pixels are assigned a score which is the Mahalanobis distance from that point to the mean of the learned model Gaussian which minimizes this distance, and whose mass is above some threshold which saves compute time by not testing individual pixels against models with very low mass. We set this threshold to 30% (of the mass of the most massive model) and found this to provide a good balance between efficiency and result quality. By using a Mahalanobis distance, we handle roads with different color and texture variation by adapting the thresholds to what constitutes drivable terrain. In this way, every point in the image (excluding sky and shadows which we have already removed from consideration) is assigned a "roadness" score. The left part of Figure 4 shows the output of this processing step.

Fig. 3. Input raw image (left) and output data of the computer vision part of the algorithm. The right image duplicates the left, overlaying the training data from the laser map (blue polygon), and the pixels identified as drivable (red area). The scene is the same as in Fig. 2.

Fig. 4. Intermediate processing steps: The left image shows the similarity of each pixel to our road model (before thresholding). The right image shows the border between ground and sky, as found by the algorithm.

E. Selecting identified patches

By thresholding away image points further away than 3σ, we get a binary image indicating the drivability of each pixel. Most roads, however, still have individual outlier pixels not classified as drivable. These pixels usually correspond to small leaves, stones, and other items not considered obstacles. To avoid being influenced by these impurities, we run a dilate filter followed by two erode filters on the image. This removes small non-drivable areas while preserving bigger obstacles and even encircling them with a non-drivable corona. Furthermore, we only accept pixels that have a connected path to the location of the training polygon. This gets rid of unwanted clutter made up from distant mountains or other small clutter that just happens to have the same properties as the main road. The resulting, final classification is visible as red pixels in the right part of Figure 3.

F. Constructing a sky-view drivability map

The next step is to re-project this 2D drivability image into a sky-view map similar to the laser occupancy map which was used to extract the training area. The major obstacle here is the absence of depth information from the image. As an approximation, we reverse the projective transform by assuming the road is flat but sloped by a constant degree upwards or downwards. This assumption is valid for almost all roads encountered in the Grand Challenge, since it allows for inclines and declines, as long as the slope does not change much. The places where this assumption is invalid usually lie in highly irregular mountainous terrain. In the mountains, the vehicle is slow anyways, so computer vision as a long-range sensor is not required for safe driving.

The road slope is accurately estimated by a low-pass filter on vehicle pitch obtained from the pose estimation system. This distinction between vehicle and road pitch allows driving inclines and declines while maintaining the ability to factor out fast camera movement, as caused by road ruts and general bumpy off-road driving.

Given 6D pose information for the camera, projection of the camera image into a vision map is straightforward. As a result, the map created by this method (right part of Fig. 2) reaches much further than the one created using only the lasers (left part of Fig. 2).

G. Incorporating the drivability map into a driving strategy

While the vision map is remarkably accurate in most driving situations, it only has two states, namely drivable and unknown cells. Simply because a part of the image does not look like the training set does not imply that it is obstacle. The laser map on the other hand is tri-state and distinguishes between obstacles and unknown areas.

For this reason, our driving strategy performs path-planning only in the laser map and instead uses the vision map as part of the speed selection strategy: By default, the vehicle has a maximum velocity of 11.2m/s (25mph). Only if the vision map contains only drivable cells among the next 40m of the planned path, the maximum velocity is set to 15.6m/s (35mph). Therefore, computer vision acts as a pre-warning system. If our vehicle is driving towards an obstacle at high speed, it is slowed down as soon as vision fails to see clear road out to 40m. Once the obstacle comes within laser range, the vehicle has already slowed down and has no problem avoiding the obstacle using the laser range finders without exceeding lateral acceleration constraints. Though not needed for the race, avoidance of stationary, car-sized obstacles in the middle of the path was demonstrated at up to 17.9m/s (40mph), on slippery gravel surface.

III. Parameter selection

A. Scoring classification results

In order to compare different algorithms and optimize parameters, we need a scoring method that enables us to evaluate the efficiency of a particular approach. Instead of tedious, hand-labeling of images as ground-truth for drivability, we chose a practical approach: Since development and testing of our vehicle was performed on last year's Grand Challenge course, we decided to train on the corridor data associated with that race. To comply with California law, DARPA had to provide participating teams with a route file that describes the allowed driving corridor and extends no wider than the actual roads width (at least for the Californian segment of the race). To make the race feasible for truck-based entrants, DARPA had to also make sure that this corridor is not much smaller than the road. Therefore, we already have a ground-truth labeling. Due to Stanley's pose estimation error following the course, this labeling is of course not always entirely accurate, but in good, open sky sections 20cm-accurate localization is obtained and in worst-case conditions about 2m error. Fortunately, the error is independent of algorithmic choices therefore not biased towards certain parameters. So comparisons of algorithms using this ground-truth are valid while the absolute performance numbers may be slightly inaccurate. Furthermore, the DARPA labeling is specified in the sky-view map instead of the image, which allows scoring in the final drivability map produced by the algorithm and not in the image, which is a mere intermediate product. Based on the corridor, our scoring function counts the number of drivable cells inside the corridor that are correctly identified as road (true positives, TP) and the number of cells outside the corridor that are wrongly classified as drivable (false positives, FP). While not all cells outside the corridor are non-drivable, they are usually not road and hence desirable to be not marked so. To make our scoring even more meaningful, we only scored cells between 25m and 50m in front of the robot, as that is the target range for vision-based sensing. The closer range is already covered by laser range finders and looking out farther is not necessary at the speeds driven in the Grand Challenge.

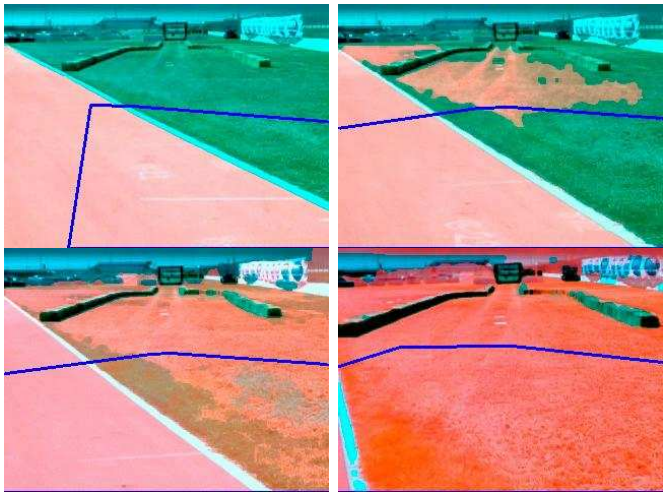

Fig. 5. These 4 images were acquired within a second during the Grand Challenge National Qualifying Event (NQE) and show the process of adaption to different terrain types. The algorithm initially learns to identify grey asphalt as road (top left), but as the training area contains more and more green grass, the algorithm switches to accept this grass (bottom right).

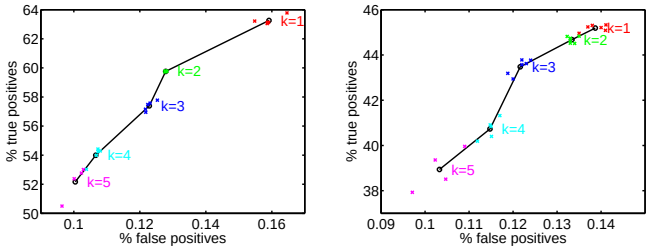

Fig. 6. The impact of k on tracking performance. The black line shows averages over different values of $n \in \{6, 7, 8, 10, 15\}$, which are plotted in clusters colored by value of $k \in \{1 \ldots 5\}$ ($\phi = 1.0$ is constant). Left and right plots are for two subsets of the Grand Challenge 05 data, each 1000 image frames in length. As expected, an increasing k leads to a tighter fit of the road model to the training data, which results in a reduction of both false and true positives.

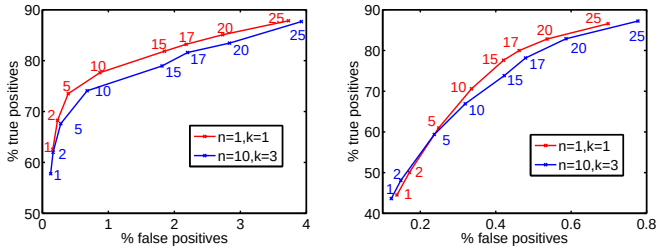

Fig. 7. The impact of ϕ on tracking performance. Increasing the parameter ϕ while keeping other parameters constant increases the overall number of cells identified as road. Left and right plot are for the same datasets as in Figure 6.

In order to compare TP and FP rates, we chose to plot Receiver Operating Characteristic (ROC) curves.

B. Further testing results

During our 2-month testing phase before the race, the vision system was confronted with a number of different obstacle types including parked cars, black and white trashcans, and a PVC tank trap model - all of which we were able to detect. During reliability testing, Stanley drove 200 autonomous miles on a 2.2-mile loop course with two white trashcans per loop, placed on light grey gravel in a 35mph speed zone. Stanley managed to avoid 180 out of the 181 obstacles and only sideswiped the remaining one. The main limitation of the system is it's color-focused design and thus the inability to detect objects with little color difference to the underlying road. A brown object on a brown dirt road without significant shadow is undetectable.

C. Optimal number of Gaussians

A major design choice in our algorithm is how many Gaussians to select in k-means clustering (k). The model storage capacity n is less important, as long as $n > k$. As n is increased, the number of frames where more than a few of the n models have a mass above the threshold is very small, so that the effect of increasing n is negligible.

Choosing a small value of k has the danger of not being able to model multi-colored roads (like pavement with lane markers), leading to a loose fit that includes colors between the ones in the training set. Choosing a large value on the other hand can lead to loss of generality and overfitting the training patch to an extent where too little actual road is found.

Figure 6 provides ROC curves for different values of these parameters. The mentioned tradeoff between small and large k is clearly visible.

D. Balancing road and obstacle finding

The other obvious place to tune TP vs. FP rate is by setting the threshold for the maximum mahalanobis distance between pixels and the MOG is accepted in section 2.4. Testing the algorithm on different kinds of road, however, unearthed a problem with very uniformly colored road surfaces. Since we do not incorporate textures in our algorithm, our camera was set to not enhance edges, and consequentially color variation in some training areas can be very low - less than one brightness value standard deviation. Even the slightest color variation further out (for example caused by CCD blooming in desert light, or just lens imperfections) would cause the road in this situation to be called non-drivable. Paradoxically, this meant that uniform colored road was detected with less certainty than textured road. To solve this problem and provide accurate classification in both road situations, we introduce an explicit minimum noise term $\phi\ I_{3x3}$ added to the measured training data covariance. By modifying this noise term, we can tune the ROC curves for true positives vs. false negatives and balance between constant and environment-adaptive color thresholding. Figure 7 displays ROC curves for different values of these thresholds.

IV. Grand Challenge Results

Our main result consists in successfully completing and winning the 2005 DARPA Grand Challenge. Figure 8 shows a significant overtaking maneuver during the race, and Figure 10 shows some other scenes that Stanley encountered during the race.

A. Real-time performance

For our application it is crucial that the algorithm runs in real-time and reacts predictably to unrehearsed environmental conditions. Since onboard power comes from the stock alternator that has to supply numerous other consumers as well, the computing power allocated for computer vision was that of just one Pentium M 1.6 Ghz Blade computer, comparable to a low-end laptop. Furthermore, this computing power is also used for receiving images from the ethernet camera, for calculating image statistics to adjust camera parameters, for video compression, logging on the hard disk, and for distributing the map to the path planning algorithm. Therefore, only half of that computing power is available for the actual computer vision part of the algorithm. To achieve performance of about 12fps and to keep the algorithm simple and hence more robust, we made the following parameter choices on race day:

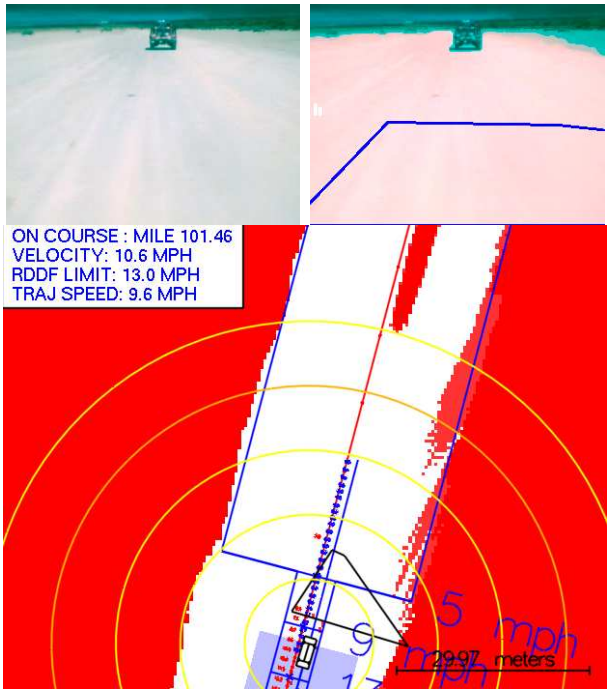

Fig. 8. Overtaking maneuver: At mile 101.5, we overtook CMU's H1ghlander vehicle. The processed image in the upper right part shows how the obstacle is clearly identified by the algorithm. The lower part shows the generated map, where H1ghlander is visible at 50m range.

- $(n, k) = (1, 1)$: Although results indicate that values of $k = 2 - 3$ and $n = 5 - 10$ perform slightly better, the algorithm is more predictable for simple parameters. Furthermore, the runtime of the pixel evaluation depends linearly on n, so smaller values increase frame rate. If the race was held a second time, or if more knowledge about the terrain types encountered had been available beforehand, we would run/have run it with values $(3, 10)$.
- Image resolution was 320×240.
- Shadow and sky removal was disabled. We chose to mount the camera and select the camera zoom such that the overwhelming proportion of the visible scene would not contain sky. Thus the computation time required to locate the horizon was not warranted by the small reduction in pixels requiring analysis.

Our code made use of the Open Source Computer Vision Library (OpenCV)[18] maintained by Intel. OpenCV is a comprehensive collection of C code optimized computer vision algorithms under a BSD license. We also utilized Intel Integrated Performance Primitives[19] software library containing assembly level optimized routines that OpenCV makes use of.

B. Impact on the race outcome

The completion time of our robot, measured by DARPA officials, was 6:53:58 hours, resulting in 1st place. Without the computer vision module, our top speed would have been 11.2m/s (25mph) instead of 15.6m/s (35mph). Figure 9 shows the average speed during the 132.2 mile race. Simulating

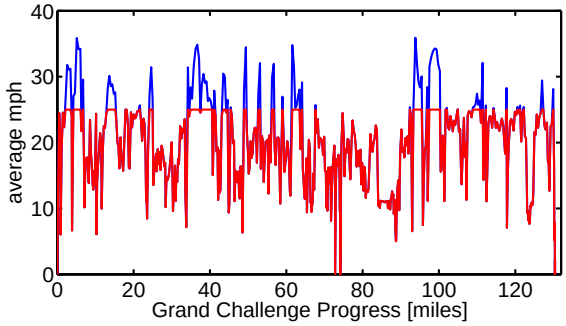

Fig. 9. Average speed (50 second window) of our vehicle during the 2005 Grand Challenge. All velocities above 11.2m/s (25mph) would not have been achieved without help of the computer vision algorithm. Please note that other factors (for example speed limits and road roughness) would also slow down the vehicle. The top speed is only achieved when all sensors indicate perfect driving conditions.

our finishing time by capping vehicle velocity at a safe laser speed of 25mph, we obtain a new finishing time of more than 7 hours and 5 minutes. The improvement seems unimpressive, but it has to be considered that the entire route had DARPA speed limits attached. In fact, post-race analysis concluded that 67.7% of the time, the DARPA speed limit or a lateral acceleration constraint limited Stanley's maximum speed. 17.6% of the time, road roughness estimated by [17] limited the speed and only 14.0% of the time, the algorithm described in this article was responsible for not driving faster. Furthermore, turning vision off would have guaranteed Stanley a second place behind CMU's Sandstorm vehicle which finished in 7:04:50 hours.

V. Conclusions

We have presented a novel method for locating roads in difficult, uneven, and inconsistent terrain. The context of the DARPA Grand Challenge involved driving 132 miles in 7 hours, starting before dawn and continuing through the diverse illumination conditions of the Nevada desert. By relying on short range sensors to train the vision system we were able to continuously learn and adapt to new conditions throughout this extended period of time. Because our method relies on fast, well developed algorithms, the overall method shows high performance in real time even on modest computer hardware. This combination of performance and robustness played a decisive role in the race victory.

While the system as-is was developed with the application of the DARPA Grand Challenge in mind, the underlying approaches can be generalized to help solve other real-world problems and finally bring Computer Vision into production systems: Systems are much more robust if they integrate multiple sensor modalities in a way that takes the advantages and disadvantages of each sensor into account. Vision should be treated as an add-on to a system, and as such it can play a role that extends capabilities significantly.

At the current development level, vision may still have failure modes if the conditions don't allow for it to operate

Fig. 10. Scenes of the 2005 Grand Challenge: The first row shows a media watch point, with the raw camera image on the left and the classification result on the right. The next rows show classification results during traversal of the mountainous Beer Bottle Pass. The blue training area varies significantly because the rugged nature of the environment causes Stanley to shake more, which in turn sometimes leaves holes in the laser map. The module that fits an area-maximized training quadrangle into this map is conservative and tries not to cover these holes. A training area that varies from frame to frame however does not harm segmentation performance, since the algorithm is executed at high frame rate and learns a road model that incorporates history.

confidently. However, systems can be designed such that they gracefully fall back to base functionality provided by other modalities. In our case, Stanley simply slowed down if vision was unable to prove road drivability out to a safe distance.

ACKNOWLEDGMENT

The authors would like to thank the members of the team, as well as the sponsors for making the project possible.

REFERENCES

[1] E. D. Dickmanns and A. Zapp, *Guiding Land Vehicles Along Roadways by Computer Vision.* Proc. AFCET Congres Automatique, Toulouse, France, October 23-25 1985.

[2] M. Lützeler and S. Baten, *Road Recognition for a Tracked Vehicle.* Proc. SPIE Conf. on Enhanced and Synthetic Vision, AeroSense 2000, Orlando, FL, USA, April 24-28 2000.

[3] R. Gregor, M. Lützeler, and E. D. Dickmanns, *EMS-Vision: Combining on- and off-road driving.* Proc. SPIE Conf. on Unmanned Ground Vehicle Technology III, AeroSense 2001, Orlando, FL, USA, April 16-17 2001.

[4] S. Baten, M. Lützeler, E. D. Dickmanns, R. Mandelbaum, and P. Burt, *Techniques for Autonomous, Off-Road Navigation.* IEEE Intelligent Systems, Vol. 13(6): 57-65, 1998.

[5] R. Gregor, M. Lützeler, M. Pellkofer, K.-H. Siedersberger, and E. D. Dickmanns, *A vision system for Autonomous Ground Vehicles With a Wide Range of Maneuvering Capabilities,* Computer Vision Systems, Second International Workshop, ICVS 2001 Vancouver, Canada, July 7-8 2001.

[6] A. Broggi, M. Bertozzi, and A. Fascioli, *ARGO and the MilleMiglia in Automatico Tour,* IEEE Intelligent Systems, Vol. 14(1): 55-65, 1999.

[7] M. Bertozzi, A. Broggi, A. Fascioli, and S. Nichele, *Stereo vision-based vehicle detection,* Proc. IEEE Intelligent Vehicles Symposium, Dearborn, MI, USA, October 3-5 2000.

[8] P. H. Batavia, D. A. Pomerleau, and C. E. Thorpe, *Predicting Lane Position for Roadway Departure Prevention,* Proc. IEEE Intelligent Vehicles Symposium, Stuttgart, Germany, October 28-30 1998.

[9] R. Aufrère, J. Gowdy, C. Mertz, C. Thorpe, C. Wang, and T. Yata, *Perception for collision avoidance and autonomous driving,* Mechatronics, Vol. 13(10): 1149-1161, 2003.

[10] F. Heimes and H.-H. Nagel, *Towards Active Machine-Vision-Based Driver Assistance for Urban Areas,* International Journal of Computer Vision, Vol. 50(1): 5-34, 2002.

[11] D. Lieb, A. Lookingbill, S. Thrun: *Adaptive Road Following using Self-Supervised Learning and Reverse Optical Flow,* Proc. Robotics Science and Systems, Cambridge, MA, USA, June 8-11 2005.

[12] H. Greenspan, J. Goldberger, and I. Eshet: *Mixture model for face-color modeling and segmentation,* Pattern Recognition Letters Vol. 22(14): 1525-1536 (2001).

[13] B. Davies and R. Lienhart, *Using CART to Segment Road Images,* Proc. SPIE Multimedia Content Analysis, Management, and Retrieval, San Jose, CA, USA, Jan 15-19 2006.

[14] S. Thrun, M. Montemerlo, and A. Aron: *Probabilistic Terrain Analysis For High-Speed Desert Driving*. Proc. Robotics Science and Systems, Philadelphia, PA, USA, August 16-19 2006.

[15] S. Ettinger, M. Nechyba, P. Ifju, and M. Waszak: *Vision-Guided Flight Stability and Control for Micro Air Vehicles,* Advanced Robotics, Vol. 17: 617-640, 2003.

[16] R. Duda and P. Hart: *Pattern Classification and Scene Analysis,* John Wiley and Sonds, New York, NY, 1973.

[17] D. Stavens and S. Thrun: *A Self-Supervised Terrain Roughness Estimator for Off-Road Autonomous Driving,* In Proc. Conference on Uncertainty in AI (UAI), Cambridge, MA, USA, July 13-16 2006.

[18] http://www.intel.com/technology/computing/opencv/index.htm

[19] http://www.intel.com/cd/software/products/asmo-na/eng/perflib/ipp/