A MULTI-RANGE VISION STRATEGY FOR AUTONOMOUS OFFROAD NAVIGATION

Raia Hadsell[1] Ayse Erkan[1] Pierre Sermanet[2] Jan Ben[2] Koray Kavukcuoglu[1] Urs Muller[2] Yann LeCun[1]

[1]Courant Institute of Mathematical Sciences
New York University
New York, USA
email:raia|ayse|koray|yann[at]cs.nyu.edu

[2]Net-Scale Technologies
Morganville, New Jersey, USA
email:psermanet|janben|urs[at]net-scale.com

ABSTRACT

Vision-based navigation and obstacle detection must be sophisticated in order to perform well in complicated and diverse terrain, but that complexity comes at the expense of increased system latency between image capture and actuator signals. Increased latency, or a longer control loop, degrades the reactivity of the robot. We present a navigational framework that uses a self-supervised, learning-based obstacle detector without paying a price in latency and reactivity. A long-range obstacle detector uses online learning to accurately see paths and obstacles at ranges up to 30 meters, while a fast, short-range obstacle detector avoids obstacles at up to 5 meters. The learning-based long-range module is discussed in detail, and field experiments are described which demonstrate the success of the overall system.

KEY WORDS
vision, learning, offroad, navigation, latency, LAGR

1 Introduction

Vision-based navigation in offroad environments is a notoriously difficult problem. The vast diversity of terrain types presents a difficult challenge for obstacle and path detection, uneven groundplane and slippery surfaces cause positioning errors, and path planning is very difficult in the absence of identifiable roadways. Traditional vision-based navigation systems use a stereo algorithm to populate traversability maps with obstacles. In the most common approach, a stereo matching algorithm, applied to images from a pair of stereo cameras, produces a *point-cloud* in which the most visible pixels are given an XYZ position relative to the robot. A traversability map can then be derived using various heuristics, such as counting the number of points that are above the ground plane in a given map cell. Maps from multiple frames are assembled in a global map in which path finding algorithms are run [6, 7, 2]. The performance of such stereo-based methods is limited, because stereo-based distance estimation is often unreliable beyond 10 or 12 meters. This may cause the system to drive as if in a self-imposed fog, driving into dead ends and taking time to discover distant pathways.

In addition to the challenges of vision, there is a dilemma posed by the competing needs of the system. In an unstructured environment, good reactivity is necessary for obstacle avoidance, since obstacles may not be recognized until they are very close. This requires a low latency system and a fast control loop. However, more advanced approaches to vision generally require high resolution images and longer processing times.

We propose a solution to the dilemma that is based on experiments performed with the LAGR (Learning Applied to Ground Robots) mobile robot platform. A multiple range architecture for perception and control is proposed which combines two navigation modules: a learning-based long-range obstacle detector, and a simpler, short-range stereo obstacle detector. The long-range module provides information about distant areas, enabling *strategic* path planning, thus avoiding dead ends and efficiently navigating toward goals. The short-range module operates at a fast frame-rate and performs *tactical* obstacle avoidance.

The long-range obstacle detector uses online learning to solve the problem of long-range obstacle and path detection. It uses dimensionality reduction and an online classifier to identify objects and paths that are far beyond the reach of stereo. Self-supervised learning, in which traversability information from one sensor is used to train a classifier on outputs from a different sensor, allows for robust performance in spite of rapidly changing terrain. In this case, we use traversability labels gleaned from the stereo module to train a classifier on image patches. The classifier can then predict the traversability costs of distant parts of the scene.

The long-range vision system has several novel features. Since it aims to generalize from the near to the far range, it must solve the perspective problem which makes far away obstacles appear smaller than close ones. We construct a distance-normalized image pyramid to cope with this issue. Secondly, the traversability labels from the stereo module can be sparse or noisy, so we maximize their usefulness by accumulating labels in a spatially-indexed quad-tree and smoothing them over time. Finally, a ring buffer is used to simulate short term memory.

Path planning is critical to a goal-driven mobile robot. The use of a multi-range navigation architecture allows for both strategic and tactical planning decisions that are merged using a ray-casting planning algorithm. The integration of the short and long range modules was tested in the field. Crashes, avoidance of dead ends, and overall time and distance to goal were used to evaluate the proposed architecture.

2 Previous Work

Considerable progress has been made over the last few years in designing autonomous off-road vehicle navigation systems. One direction of research involves mapping the environment from multiple active range sensors and stereo cameras [8, 12], and simultaneously navigating and building maps [7, 19] and classifying objects.

Estimating the traversability of an environment constitutes an important part of the navigation problem, and solutions have been proposed by many; see [4, 13, 15, 16, 21]. However, the main disadvantage of these techniques is that they assume that the characteristics of obstacles and traversable regions are fixed, and therefore they cannot easily adapt to changing environments. By contrast, the vision system presented in this paper uses online learning and is adaptable to any environment.

A number of systems that incorporate learning have also been proposed. These include ALVINN [14] by Pomerlau, MANIAC [5] by Jochem et al., and DAVE [9] by LeCun et al. Many other systems have been proposed in recent years that include supervised classification [11, 3]. These systems were trained offline using hand-labeled data which has two major disadvantages. First labeling requires a lot of human effort and secondly offline training limits the scope of the robot's expertise to environments seen during training.

More recently, *self-supervised* systems have been developed that reduce or eliminate the need for hand-labeled training data, thus gaining flexibility in unknown environments. With self-supervision, a reliable module that determines traversability can provide labels for inputs to another classifier. This is known as *near-to-far* learning. Using this paradigm, a classifier with broad scope and range can be trained online using data from the reliable sensor (such as ladar or stereo). Not only is the burden of hand-labeling data relieved, but the system can robustly adapt to changing environments on-the-fly. Many systems have successfully employed near-to-far learning in simple ways, primarily by identifying ground patches or pixels, building simple color histograms, and then clustering the entire input image.

The near-to-far strategy has been used successfully for autonomous vehicles that must follow a road. In this task, the road appearance has limited variability, so simple color/texture based classifiers can often identify road surface well beyond sensor range. Using this basic strategy, self-supervised learning helped win the 2005 DARPA Grand Challenge: the winning approach used a simple probabilistic model to identify road surface based on color histograms extracted immediately ahead of the vehicle as it drives [1]. In a slightly more complicated approach by Thrun et al., previous views of the road surface are computed using reverse optical flow, then road appearance templates are learned for several target distances [10].

Several other approaches have followed the self-supervised, near-to-far learning strategy. Stavens and Thrun used self-supervision to train a terrain roughness predictor [18]. An online probabilistic model was trained on satellite imagery and ladar sensor data for the Spinner vehicle's navigation system [17]. Similarly, online self-supervised learning was used to train a ladar-based navigation system to predict the location of a load-bearing surface in the presence of vegetation [23]. A system that trains a pixel-level classifier using stereo-derived traversability labels is presented by Ulrich [20].

Figure 1. The robot weighs 70kg, measures 1 meter high, and can reach a maximum speed of 1.3 meters/second.

3 Overview of Navigation

This section gives an overview of the full navigation system developed using the LAGR (Learning Applied to Ground Robots) robot platform (see Figure 1). Both the robot and the reference "baseline" software were built by Carnegie Mellon University and the National Robotics Engineering Center. The LAGR platform is composed of two stereo pairs of color cameras, a GPS receiver, an inertia measurement unit (IMU), wheel encoders and a front bumper. Four onboard Linux machines (2Ghz Pentium 4 equivalent with 1Gb of RAM) are connected via a 1Gb ethernet network. One computer is dedicated to path planning, one to low-level driving functions and two to the stereo pairs.

Although reference "baseline" software was provided, none was used in our system. Our navigation system consists of 5 major components, described below (see Figure 2).

Obstacle Detection. The obstacle detection module(s) use camera inputs to identify traversable and non-traversable regions. The module(s) then populate the vehicle map with the traversability information in the form of cost and confidence values. We used two obstacle detectors for this research. One is a fast, short-range stereo module (FAST-OD), and the other is a slower, long-range vision module (FAR-OD).

Vehicle Map. The vehicle map is a local map in polar coordinates that is fixed relative to the robot position. It is 100 degrees wide and has a 40 meter radius. It stores cost and confidence data which is delivered by the various obstacle detectors.

Local Navigation. The local navigation is based on the vehicle map. It determines a set of candidate waypoints based on cost, confidence, and steering requirements. The candidate waypoint is picked which lets the vehicle progress toward the goal. Driving commands are issued based on this choice.

Global Map. The global map is a Cartesian grid map into which cost and confidence information from the vehicle map is copied after each processed frame. The global map

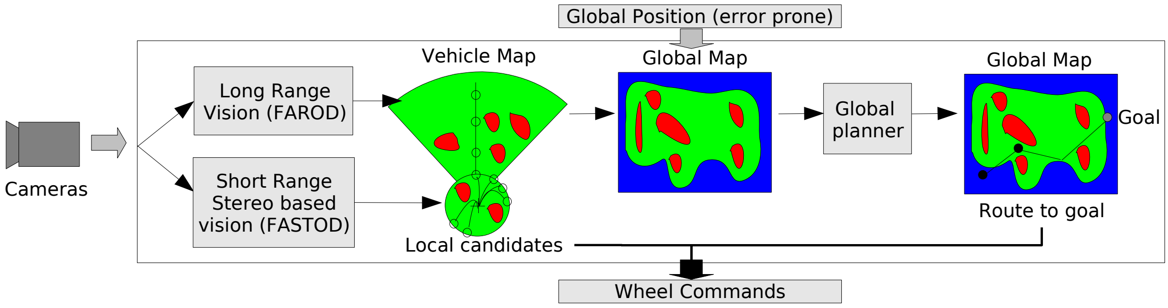


Figure 2. A flow chart of our navigation system. The long-range and stereo obstacle detectors both populate the vehicle map, where local navigation is done. The local map gets written to the global map after every frame, where route planning is done with the global planner.

is the system's "memory".

Global Planner. The global planner finds a route to the goal in the global map, starting with candidate points proposed by the local navigation module. The algorithm is a modified A* algorithm which operates on rays rather than grid cells.

This system architecture is designed to make the robot robust against positioning errors. The vehicle POSE (POSition Estimation), as computed by an onboard IMU (inertia measurement unit), is prone to error from wheel slip and drift. In the absence of visual odometry, this can cause errors in the global map. The strategy of the system, however, is to "trust our eyes" and issue driving commands directly from the vehicle map, which is not affected by POSE error.

4 FAST-OD and FAR-OD: A Multi-Range Architecture

This section describes the multiple range architecture of the navigation system. To maximize the usage of both long and short range obstacle detectors, each is implemented as an independent process running at its own pace. This allows the short-range module to use a lower image resolution and further decrease its control loop period. This architecture has its basis in our understanding of human vision. Research has shown that human subjects focus on nearby objects much more frequently than distant areas: occasional distant gazing suffices to maintain a global trajectory, but frequent nearby gazing is necessary for obstacle avoidance[22].

The short-range module is termed FAST-OD (Fast Obstacle Detection), since its primary function is to provide quick obstacle avoidance. The long-range module is termed FAR-OD in accordance with its primary purpose: accurate, long-range vision. When both processes are run on the same CPU, the FAST-OD is given priority. This is done by giving control to the FAST-OD when a new frame arrives. The two processes must be balanced according to the available CPU budget, running speed, and visual distance to achieve the desired behavior. One could extend this idea by giving more CPU cycles to FAR-OD when going straight and more to FAST-OD when the robot begins turning.

In our current system, FAST-OD uses stereo input images at a resolution of 160x120 to produce a traversability map with a radius of 5 meters. When run alone on the robot CPU, it reaches 10Hz and a latency from the acquired image to the wheel commands of about 300 ms. The FAR-OD uses a stereo module at input resolution 320x240 as training input to an online/offline classifier. When run together with a reasonable CPU trade-off between the modules, FAST-OD has a frequency of about 4Hz and latency of 300ms and FAR-OD has a frequency between 0.5Hz and 1Hz.

5 FAR-OD: Long-range Vision through Self-Supervised Online Learning

This section discusses the long-range vision system. We first describe the image processing: estimating the groundplane, leveling the horizon, and constructing a distance-normalized pyramid. Then the learning process is discussed: labeling, training, and classifying.

5.1 Image Processing: Horizon Leveling and Distance Normalization

Vision-based navigation classifiers commonly use color/texture histograms built from small image patches, thus limiting the classifier to color and texture discrimination. However, it is beneficial to use larger windows from the image, thus providing a richer context for more accurate training and classification. Recognizing the feet of objects is critical for obstacle detection, and the task is easier with larger windows.

There is an inherent difficulty with training on large windows instead of color/texture patches. In image space, the apparent size of obstacles, paths, etc. varies inversely with distance, making generalization from near-range to far-range unlikely. Our approach deals with this problem by building a *distance-invariant pyramid of images at multiple scales*, such that the height of similar objects is consistent, regardless of different distances from the camera. We also warp the image to level the horizon so that the normalization is consistent in spite of uneven terrain.

In order to build a pyramid of horizon-leveled sub-images, the ground plane in front of the robot must first be

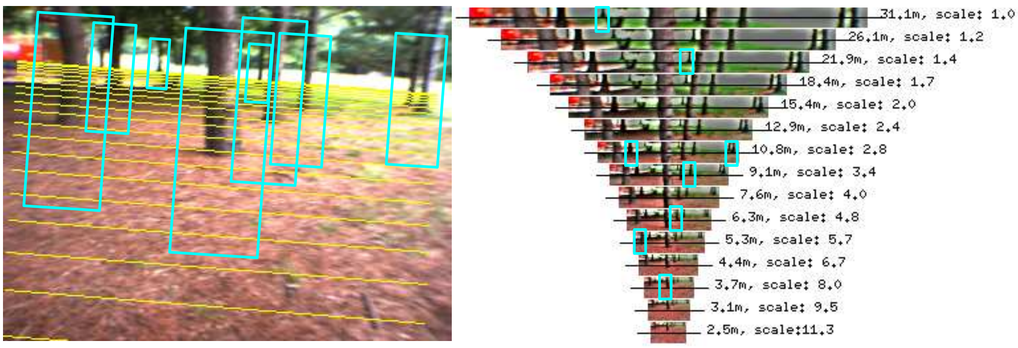


Figure 3. Sub-images are extracted around imaginary lines on the ground that are computed using the ground plane estimate. The boxes on the left correspond to the boxed locations in the pyramid on the right. The size of each sub-image on the left is determined inversely by the distance of the footline from the camera. All the sub-images are subsampled to a uniform height (20 pixels). The distance-normalized image pyramid is shown, containing 24 bands.

estimated by performing a robust fit of the point cloud obtained through stereo. A Hough transform is used to produce an initial estimate of the plane parameters. Then, a least-squares fit refinement is performed using the points that are within a threshold of the initial plane estimate.

To build the image pyramid, differently sized sub-images are cropped from the original image such that each is centered around an imaginary *footline* on the ground. Each footline is a predetermined distance (using a geometric progression) from the robot's camera, and that distance determines the size of the cropped sub-image. For [*row, column, disparity, offset*] plane parameters $P = [p_0, p_1, p_2, p_3]$ and desired disparity d, the image coordinates (x_0, y_0, x_1, y_1) of the footline can be directly calculated.

After cropping a sub-image from around its footline, the sub-image is then subsampled to make it a uniform height (12 pixels), resulting in image bands in which the appearance of an object on the ground is independent of its distance from the camera (see Figure 3). These uniform-height, variable-width bands form a distance-normalized image pyramid whose 24 scales are separated by a factor of $2^{\frac{1}{4}}$.

5.2 Learning: Labeling, Training, and Classification

The long-range obstacle detector is trained using data from every frame it sees. The distance-normalized pyramid described in the previous section is divided into overlapping identically-sized windows (20x11 pixels). Some of these windows can be associated with stereo labels. The stereo module can provide 3 types of labels: *ground*, *obstacle*, and *blocked*. Windows labeled as ground contain disparity points that are near to the goundplane. Windows labeled as obstacle contain disparity points that are above the groundplane. Windows labeled as blocked contain disparity points that belong to obstacles in front of that window. Furthermore, each associated window and label is entered into a spatially-indexed quad-tree. All labels that are entered for a particular XYZ location are accumulated and translated into a traversability probability: $\frac{gr}{ob+gr}$, where gr is the number of ground labels and ob is the number of obstacle labels.

The windows must be mapped to a lower dimensional space before they can be used for training an online classifier. This is done using a trained convolutional neural network. The network is trained offline using a diverse collection of 31,600 stereo-labeled images taken from 158 logfiles. The output of the network is a 240 dimension feature vector.

The online classifier is a logistic regression trained with cross-entropy loss. It has 723 trainable parameters and 3 outputs that represent the 3 possible classes (ground, obstacle, or blocked). Weight decay is used to regularize the regression towards a set of previously learned default parameters. Learning is done by stochastic gradient descent, providing a natural generalization over successive frames.

Since the training is discriminative, the classifier's performance can be distorted if it is "swamped" with training samples from one class. Therefore a ring buffer is used to balance the samples of each class and provide a simple short-term memory. The ring buffer holds a set number of samples from each class, and is refreshed on every frame.

After training, the parameters are fixed and the entire pyramid of windows is classified by a simple inference. Training and classification are very fast, taking less than 100ms per frame.

6 Planning in a Multi-Range Architecture

The path planner in a multi-range navigation system must integrate different sources of traversability information that arrive at different times. The FAST-OD and FAR-OD modules in our system each produce a traversability (cost) map, but they are updated at different speeds and they provide cost estimates for different ranges (0 to 5 meters vs. 5 to 35 meters). The path planning algorithm should run every time a new map comes from the FAST-OD, making it necessary to affinely transform the older FAR-OD map according to the robot's movement. It can then be combined with the new FAST-OD cost map.

The route chosen by the path planner must satisfy two requirements: it must avoid collisions with obstacles, and it must navigate towards the goal. Since the first requirement is more immediate than the second, we use the following planning protocol. We identify desirable trajectories in the FAR-OD map, then use these to influence the allowable trajectories given by the FAST-OD map. This arrangement allows us to modulate the contribution of the FAR-OD.

An advantage of separating the FAST and FAR-OD planning processes is that we can easily use different planning algorithms for each map. For example, planning is currently done by casting rays in each direction and stopping when an obstacle is encountered. A future area of work will be to integrate the vehicle's dynamics in the planner by casting curves instead of rays in the map. This algorithm does not need to be used at more than 5 meters radius with our robot, thus we can keep this high dimensional algorithm small enough to be efficient and maintain the FAR-OD planner at the same time.

7 Results

The proposed system was tested in the field in order to compare the performance of the long-range vision in various environments and to compare the performance of the FAST-OD and combined FAST-OD + FAR-OD configurations. Two courses were used in testing. Course 1 had a Y-shaped path with a dead end, and Course 2 was an L-shaped path that followed an irregular border of sparse scrub and shrubbery.

Course 1 presented a choice of 2 paths that could potentially reach a goal 100 meters distant. The left path was a dead end that ended at a building, and the right path was a narrow gravel road (see Figure 4). Each run was terminated when the robot crossed a finish line on the gravel road, rather than proceeding all the way to the goal. Course 2 presented a different challenge. Because of sparse foliage and repeating patterns, the stereo module persistently detected openings through the bushes, even though the bushes were definitely non-traversable. This course tested the ability of the long-range module to accurately assign high cost to the hazardous scrub border and avoid getting stuck.

The robot was run with (1) only FAST-OD, (2) FAST-OD + FAR-OD without initializing the classifier parameters, and (3) the complete system with initialized parameters. Each configuration was tested 5 times. The classifier parameters were initialized by driving briefly around the environment. Performance results on both test courses are shown in Table 5. The time to goal, distance traveled to goal, and number of crashes all improve substantially when the FAR-OD is used. In addition, on Course 1, the FAST-OD configuration enters the dead end on every run, whereas the FAR-OD configuration avoids the dead end.

A global map is built by the system as it traverses a course. The maps for two test runs (on Course 1), overlaid on satellite images of the site, are shown in Figure 6.

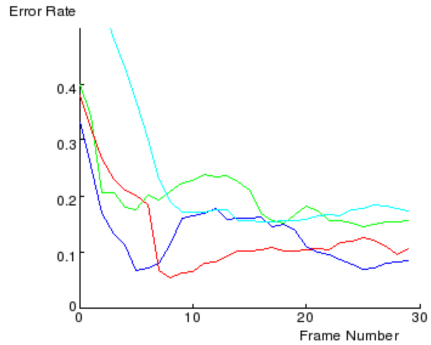


Figure 8. The plot shows the error rate for 50 consecutive frames on each of 5 runs on Course 2. The error rate begins at $> 20\%$, then quickly drops and hovers at $5\% - 20\%$.

The short-range FAST-OD cannot see the building in time to avoid the dead end. Figure 7 shows selected frames processed by the FAR-OD module.

A plot of the online classifier error rate (on the validation set, not the training set) is shown for the first 50 frames for each of the 5 runs on Course 2 (see Figure 8). At a fixed learning rate of 0.0001, the learning begins with a high error of $> 20\%$ and quickly descends towards a lower "running" error rate of $5 - 20\%$.

8 Conclusions

We have presented an autonomous navigation system with a multi-range architecture that includes a reactive, low-latency obstacle avoidance module and a learning-based long-range vision module. The long-range vision uses self-supervised online learning to predict the traversability of areas up to 30 meters away. Experimental results confirm that the long-range module is able to detect and avoid dead ends, while the short-range module myopically explores them. Although robotic navigation in unstructured, outdoor environments remains an extremely difficult, unsolved problem, our system gives hope for one day attaining human-level understanding of visual scenes.

Acknowledgments

This work was supported in part by DARPA under the LAGR program.

References

[1] H. Dahlkamp, A. Kaehler, D. Stavens, S. Thrun, and G. Bradski. Self-supervised monocular road detection in desert terrain. In *Proc. of Robotics: Science and Systems (RSS)*, June 2006.

[2] S. B. Goldberg, M. Maimone, and L. Matthies. Stereo vision and robot navigation software for planetary exploration. In *IEEE Aerospace Conf.*, March 2002.

Figure 4. The testing site. *Left:* the robot is at the start point, facing the goal, which is 100m away, beyond the trees. The left path is blocked by a large building. The right path is clear. *Right:* the robot is stuck in brush when it runs without FAR-OD to guide it away from the dead end.

Test Mode Course 1	**Runs**	**Average Time**	**Average Distance**	**Average Crashes**	**Average Dead end**
(1) FAST-OD only	5	97.84 sec	42.78 meters	0.8	1.0
(2) FAST-OD+FAR-OD (untrained)	5	81.04 sec	34.7 meters	0.0	0.0
(3) FAST-OD+FAR-OD (trained)	5	**53.08 sec**	**23.4 meters**	0.0	0.0
Course 2					
(1) FAST-OD only	4	180.77 sec	66.1 meters	1.3	–
(3) FAST-OD+FAR-OD (trained)	4	**133.8 sec**	**59.5 meters**	0.0	–

Figure 5. Results of field test on two courses. Mode 1: the FAST-OD goes into the dead end on every run, and gets stuck in brush often (stereo is poor at detecting sparse branches and grass). Mode 2: when the FAR-OD is not first initialized, it performs poorly, only seeing the dead end on half the runs. Mode 3: when the FAR-OD is first initialized by brief driving, it performs well, never entering the dead end and never crashing.

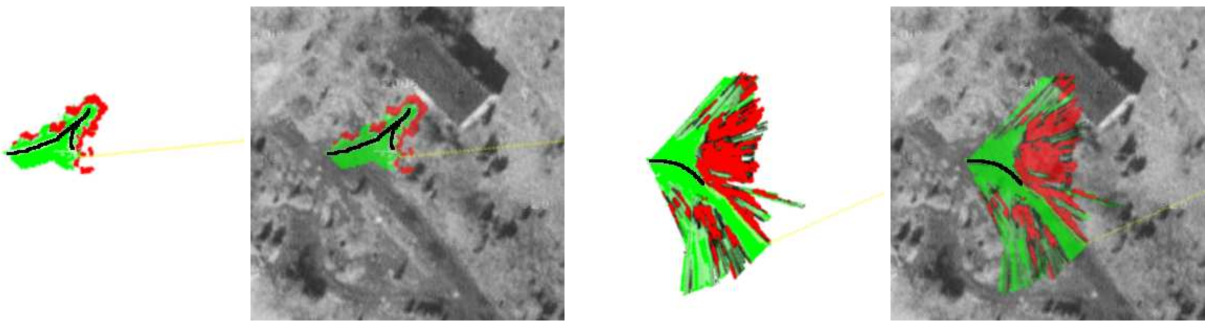
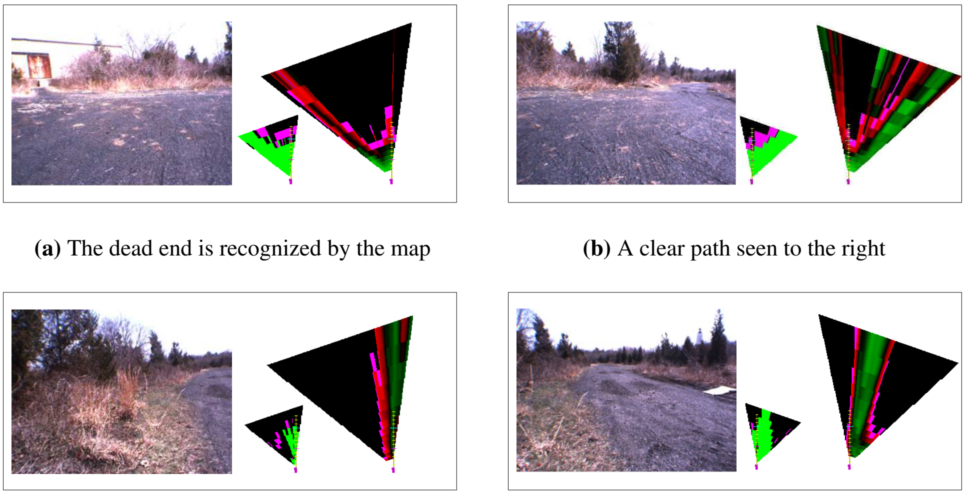


Figure 6. Global maps from the field test, overlaid on satellite images of the Course 1 test site. The left maps show the short-range map created by the FAST-OD, and the right maps show the longer range maps created by the FAR-OD. The black line indicates the path of the robot as it traversed the course. The FAST-OD explores the dead-end before finding the road, and the FAR-OD sees the dead-end and avoids it.



(a) The dead end is recognized by the map

(b) A clear path seen to the right

(c) The left side of the road

(d) The right view of the road

Figure 7. The performance of the long-range vision on the field test. Each example shows the input image (left), the stereo labels (middle), and the map produced by the long-range vision (right). *Top row:* Near the beginning of the test, the dead end is seen to the left and the clear road is seen to the right, so the dead end is avoided. *Bottom row:* At the end of the test, the robot clearly sees the road far beyond stereo range.

[3] T. Hong, T. Chang, C. Rasmussen, and M. Shneier. Road detection and tracking for autonomous mobile robots. In *Proc. of SPIE Aeroscience Conference*, 2002.

[4] A. Huertas, L. Matthies, and A. Rankin. Stereo-based tree traversability analysis for autonomous off-road navigation. In *Workshop of Applications of Computer Vision*. IEEE, 2005.

[5] T. M. Jochem, D. A. Pomerleau, and C. E. Thorpe. Vision-based neural network road and intersection detection and traversal. In *Proc. of Int'l Conf on Intelligent Robots and Systems (IROS)*, volume 03, pages 344–349. IEEE, 1995.

[6] A. Kelly and A. Stentz. Stereo vision enhancements for low-cost outdoor autonomous vehicles. In *Int'l Conf. on Robotics and Automation, Workshop WS-7, Navigation of Outdoor Autonomous Vehicles*, May 1998.

[7] D. J. Kriegman, E. Triendl, and T. O. Binford. Stereo vision and navigation in buildings for mobile robots. *Trans. Robotics and Automation*, 5(6):792–803, 1989.

[8] E. Krotkov and M. Hebert. Mapping and positioning for a prototype lunar rover. In *Proc. of Int'l Conf. on Robotics and Automation (ICRA)*, pages 2913–2919. IEEE, May 1995.

[9] Y. LeCun, U. Muller, J. Ben, E. Cosatto, and B. Flepp. Off-road obstacle avoidance through end-to-end learning. In *Advances in Neural Information Processing Systems (NIPS)*. MIT Press, 2005.

[10] D. Leib, A. Lookingbill, and S. Thrun. Adaptive road following using self-supervised learning and reverse optical flow. In *Proc. of Robotics: Science and Systems (RSS)*, June 2005.

[11] R. Manduchi, A. Castano, A. Talukder, and L. Matthies. Obstacle detection and terrain classification for autonomous off-road navigation. *Autonomous Robot*, 18:81–102, 2003.

[12] L. Matthies, E. Gat, R. Harrison, B. Wilcox, R. Volpe, and T. Litwin. Mars microrover navigation: Performance evaluation and enhancement. In *Proc. of Int'l Conf on Intelligent Robots and Systems (IROS)*, volume 1, pages 433–440. IEEE, August 1995.

[13] R. Pagnot and P. Grandjean. Fast cross country navigation on fair terrains. In *Proc. of Int'l Conf. on Robotics and Automation (ICRA)*, pages 2593 –2598. IEEE, 1995.

[14] D. A. Pomerlau. Knowledge based training of artificial neural networks for autonomous driving. *J. Connell and S. Mahadevan, eds., Robot Learning*, 1993.

[15] A. Rieder, B. Southall, G. Salgian, R. Mandelbaum, H. Herman, P. Render, and T. Stentz. Stereo perception on an off road vehicle. *Intelligent Vehicles*, 2002.

[16] S. Singh, R. Simmons, T. Smith, A. Stentz, V. Verma, A. Yahja, and K. Schwehr. Recent progress in local and global traversability for planetary rovers. In *Proc. of Int'l Conf. on Robotics and Automation (ICRA)*, pages 1194–1200. IEEE, 2000.

[17] B. Sofman, E. Lin, J. Bagnell, N. Vandapel, and A. Stentz. Improving robot navigation through self-supervised online learning. In *Proc. of Robotics: Science and Systems (RSS)*, June 2006.

[18] D. Stavens and S. Thrun. A self-supervised terrain roughness estimator for off-road autonomous driving. In *Proc. of Conf. on Uncertainty in AI (UAI)*, 2006.

[19] S. Thrun. Learning metric-topological maps for indoor mobile robot navigation. *Artificial Intelligence*, 99(1):21–71, February 1998.

[20] I. Ulrich and I. R. Nourbakhsh. Appearance-based obstacle detection with monocular color vision. In *Proc. of Conf. of the Amer. Assoc. for Artificial Intelligence (AAAI)*, pages 866–871, 2000.

[21] N. Vandapel, D. F. Huber, A. Kapuria, and M. Hebert. Natural terrain classification using 3d ladar data. In *Proc. of Int'l Conf. on Robotics and Automation (ICRA)*. IEEE, 2004.

[22] M. Wagner, J. C. Baird, and W. Barbaresi. The locus of environmental attention. *J. of Environmental Psychology, 1:195-206*, 1980.

[23] C. Wellington and A. Stentz. Online adaptive rough-terrain navigation in vegetation. In *Proc. of Int'l Conf. on Robotics and Automation (ICRA)*. IEEE, April 2004.