Adaptive Long Range Vision in Unstructured Terrain

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

(1) Courant Institute of Mathematical Sciences
New York University
New York, NY USA

(2) Net-Scale Technologies
Morganville, NJ USA

***Abstract*— A novel probabilistic online learning framework for autonomous offroad robot navigation is proposed. The system is purely vision-based and and is particularly designed for predicting traversability in unknown or rapidly changing environments. It uses self-supervised learning to quickly adapt to changing terrains after processing a small number of frames, and it can recognize terrain elements such as paths, man-made structures, and natural obstacles at ranges up to 30 meters. A convolutional neural network, trained offline, is used to extract discriminative features for the online classifier. The system is developed on the LAGR mobile robot platform and the performance is evaluated using multiple metrics, including ground truth information.**

I. INTRODUCTION

Autonomous off-road robot navigation in unknown environments is a challenging task. One major difficulty is the detection of obstacles and traversable areas when no prior information about the terrain is known. Long range vision is crucial, especially for robotics tasks where efficient goal-driven planning and driving is the goal. Depending on image resolution and processor speeds, stereo algorithms are generally only accurate up to 10 to 12 meters, whereas in open land, camera images contains information far beyond that. This work focuses on conveying the short range knowledge of the environment to long range vision via self-supervised near-to-far learning.

The diversity of the terrain and the lighting conditions of outdoor environments make it difficult to employ a database of obstacle templates or features, or other forms of pre-defined static description collections, which in turn necessitates the use of machine learning techniques. This paper proposes the use of learned adaptation skills for autonomous robots in offroad outdoor courses, which have high variability in visual content.

The learning architecture comprises two parts, an offline-trained feature extraction module that provides an initial notion of the world in the form of discriminative feature vectors. The feature extractor is trained offline on many environments that the robot was previously exposed to, and an online learning module that enables adaptation to any new, unseen terrain. The proposed system does not require any human intervention or labeling at any level, which is an advantage in terms of practicality and implementation concerns.

The proposed approach was developed as part of the navigation framework on the LAGR (Learning Applied to Ground Robots) robot platform. For details of the LAGR program and platform, see [1].

II. PREVIOUS WORK

Statistical learning techniques have been used to improve autonomous navigation systems for a decade or more. These early systems, including ALVINN [13] by Pomerlau, MANIAC [5] by Jochem et al., and DAVE [9] by LeCun et al.; use supervised learning to map visual input to steering angles. Many other systems have been proposed that rely on supervised classification [12], [4]. These systems are trained offline using hand-labeled data. Hand labeling, unfortunately, requires a lot of human effort and offline training limits the scope of the robot's expertise to environments seen during training.

To overcome these limitations, navigation systems have been developed that are capable of learning traversability labels directly from the environment. They do this through *self-supervision*: a reliable sensor provides traversability information that is then learned, either online or in batch mode at intervals, by a classifier that operates on data from another, less reliable sensor. Not only is the burden of hand-labeling relieved, but the system becomes flexible to new environments. This strategy has proved especially useful for road-following systems, which can use simple color and texture information to track the shape and position of a road. Self-supervised learning helped win the 2005 DARPA Grand Challenge: the winning team used a simple probabilistic model to identify road surface based on color histograms extracted immediately ahead of the vehicle as it drives [3]. 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, learning strategy. Stavens and Thrun used self-supervision to train a terrain roughness predictor [15]. An online probabilistic model was trained on satellite imagery and ladar sensor data for the Spinner vehicle's navigation system [14]. 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 [17]. A system that trains a pixel-level classifier using stereo-derived traversability labels is presented by Ulrich [16]. Recently Kim et al. [6] proposed an autonomous offroad navigation system that estimates traversability in an unstructured, unknown outdoor environment.

The proposed system incorporates feature extraction and label propagation into an online learning framework that is designed for maximum flexibility and adaptability in changing, offroad environments.

III. OVERVIEW OF THE SYSTEM

As mentioned previously, the proposed long range obstacle detection system comprises (LROD) two parts, a feature extractor that is used primarily for dimensionality reduction and to derive the more discriminative information in the data, and an online module that learns the traversablity of the terrain using the stereo labels in an adaptive manner. The feature extraction is done with a multi-layer convolutional network trained offline with data from log files captured in various environments. The features are then used as inputs to the online module as the robot traverses a course.

For each pair of stereo images received, the long range module does a series of procedures, including pre-processing, feature extraction, training, and classification steps. The steps in one full processing cycle are listed in Table I, along with the average processing time for each step. Section IV discusses the image pre-processing and the feature extraction, and Section VI describes the online label propagation and training strategies. The proposed approach was tested using two complementary evaluation measures, and results are presented in Section VII.

IV. IMAGE PRE-PROCESSING

On every processing cycle, the long range module receives a pair of stereo color images at a resolution of 320x240. In order to train a classifier on these images, the visual data in the images must be transformed into discrete windows of information and the windows must be labeled with a traversability value. This section describes the pre-processing and labeling steps.

A. Ground Plane Estimation

The first step is to rectify the images and run a stereo algorithm, the result of which is a point cloud in RCD (row, column, disparity) space: $P = (r_1, c_1, d_1), (r_2, c_2, d_2), ..., (r_n, c_n, d_n)$. The “Triclops” stereo algorithm was developed by Point Grey Research, the vendors of the Bumblebee cameras used by the robot. From the point cloud P, the ground plane can be estimated: a necessary step for assigning traversability labels. The ground plane is estimated initially using a Hough transform, then refined using a PCA refit on the points that are within a threshold of the initial plane. Finding a ground plane allows us to map pixels in the image to XYZ locations in the real world and to determine their distance from the plane. The ground plane is thus the basis of much of our processing, allowing computation of stereo labels, correspondance of image data and real world coordinates, distance/scale normalization, and horizon leveling.

B. Contrast Normalization

The input image is converted to the YUV color space and normalized. The U and V color channels are normalized using an individual mean and variance for each channel, but the Y channel, which contains the luminance information, is normalized over small neighborhoods in order to protect texture and image information while alleviating the effect of dark shadows and bright sunlight. Pixel x in image I is normalized by the values in a soft window centered on x:

$$x = \frac{x}{\sum_{y \in I^W, k \in K} yk}$$

, where I^W is a 16x16 window in I and K is a smooth, normalized 16x16 kernel.

C. Horizon Leveling and Scale-Normalized Pyramid

Image pyramids have been used for image processing for decades (see [2]), and more recently have been used for scale-invariant object recognition (see [11]). We developed a pyramid-based approach to the problem of distance and scale in images. The classifier is expected to generalize from near-range image windows to long-range image windows, but this is extremely difficult because of the effect of distance on scale.

Our solution is to build a distance-normalized image pyramid by extracting sub-images at different target distances in the image and subsampling them to a uniform height. The result is that similar obstacles in the image (e.g., a tree at 10 meters away and a similar tree at 30 meters away) appear in different rows in the pyramid at a similar scale (e.g., both trees are 12 pixels high), making it easier to generalize from one to the other (see Figure 1). Each pyramid row is centered around an imaginary *footline* on the ground that is a fixed distance from the robot. There are 24 footlines and corresponding pyramid rows: their distances form a $2^{\frac{1}{6}}$ geometric progression, with the closest at 0.5 meters and the furthest at 30 meters. The rows have a uniform height of 20 pixels and a width that varies from 36 pixels to 300 pixels.

D. Stereo Labeling

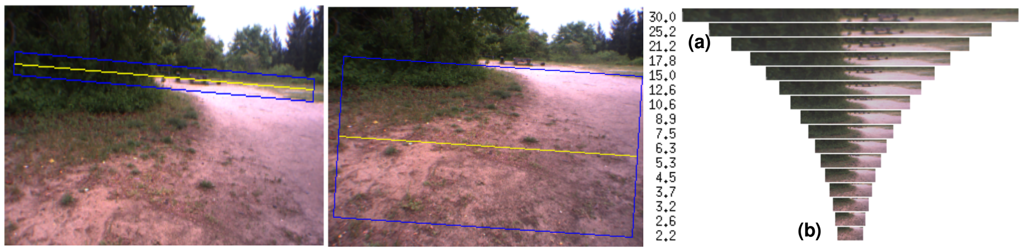
The stereo algorithm produces a point cloud of RCD values (*row, column, disparity*), and the distance of each point from the ground plane can be computed once the parameters of the plane have been estimated. Building a traversability map from these points can be done with simple heuristics. The points are collected in bins that correspond to the real world coordinates of windows in the pyramid. Heuristics are used to decide whether each bin is a traversable (*ground*) or non-traversable (*obstacle*) location. The bin could also be given a label of *blocked*, if there is a nearby obstacle that occludes that bin's location. Some windows in the pyramid can thus be labeled (ground, obstacle, or blocked) according to the RCD points at the footline of the window (see Figure 2). After stereo labeling, the pyramid windows have labels as follows:

1) *obstacle*: There is an obstacle at the footline of the window.

TABLE I

OVERVIEW OF PRINCIPLE PROCESSING STEPS IN THE VISION SYSTEM.

Ordering	Processing Step	Processing Time
	Pre-processing	
1	Image rectification and point cloud extraction	45 ms
2	Ground plane estimation	43 ms
3	Conversion to YUV and normalization	67 ms
4	Horizon leveled, distance-normalized pyramid	60 ms
	Labeling	
5	Stereo labeling of windows in pyramid	20 ms
	Feature Extraction	
6	Feature extraction (convolutional neural network)	385 ms
	Label Propagation	
7	Query quad-tree for matching windows	1 ms
8	Label query results with probabilistic labels	0 ms
9	Insert feature vectors into quad-tree	0 ms
10	Add labeled samples to ring buffer	1 ms
	Online Training and Classification	
11	Train logistic regression on ring buffer contents	55 ms
12	Classify all windows in pyramid using trained regression	12 ms
	Total	653 ms



(a). sub-image extracted from far range. (21.2 m from robot). **(b).** sub-image extracted at close range. (2.2 m from robot). **(c).** the pyramid, with rows (a) and (b) corresponding to sub-images at left.

Fig. 1. Sub-images are extracted according to imaginary lines on the ground (computed using the estimated ground plane). *(a)* Extraction around a footline that is 21m away from the vehicle. *(b)* Extraction around a footline that is 1.1m away from the robot. The extracted area is large, because it is scaled to make it consistent with the size of the other bands. *(c)* All the sub-images are subsampled to 20 pixels high.

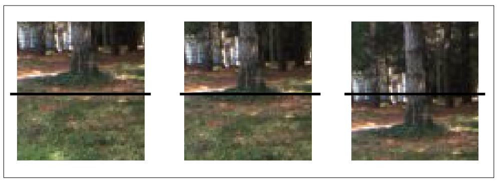


Fig. 2. The 3 possible labels: *left*: the footline is on open ground, so label = ground; *center*: the base of the object is on the footline, so label = obstacle; *right*: the footline is blocked by a nearby object, so label = blocked.

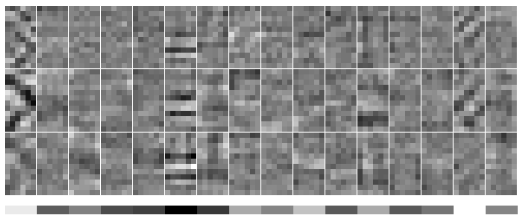


Fig. 3. The kernels learned by the feature extractor using offline training show a sensitivity to horizontal boundary lines

2) *blocked*: The footline is occluded by some object in front of it.
3) *ground*: The location corresponding to the window's footline is traversable.

V. FEATURE EXTRACTION

A classifier can't be trained online using large color windows: the computational burden would be too high for runtime processing. In addition, we want to extract discriminative features from the input windows. Therefore, a convolutional neural network is used to extract features from the color input windows.

Convolutional neural networks have been used successfully in areas such as speech and handwritten character recognition; see [8], [7]). A convolutional network contains local receptive fields that are trained to extract local features and patterns. It also contains subsampling, so that detected features are pooled. The architecture of a convolutional network makes it naturally shift and scale invariant, so it is ideal for learning visual features for navigation.

The convolutional network trained for feature extraction has two convolutional layers and one subsampling layer. The first convolutional layer has 48 7x6 filters, shown in Figure 3, and the second layer has 240 filters. The filters show that the network is very responsive to horizontal structures, such as obstacle feet and other visual boundaries. The network was trained on a data set obtained by computing labels over a diverse set of 130 logfiles.

The network produces feature vectors with 240 features.

VI. ONLINE LEARNING

Throughout this paper, online learning is used to refer to the process of learning the long-range (up to 30m) traversability information of the terrain in real-time from the stereo during the robots course. More specifically, the long range module learns how to discriminate the input windows in the distance normalized and scale invariant pyramid as one of three classes, obstacle, traversable or blocked.

At every video processing cycle, a traversability label is associated with each window in stereo range and stored in a quadtree data structure according to its XYZ coordinates in the robot's local coordinate system. Therefore, as the robot proceeds in its environment it collects features and corresponding stereo labels in this map.

From this collection, soft-labels for the pyramid windows are calculated as the ratio of each label in the quadtree cell corresponding to that window. This, in turn, softens the classification decision boundary and eliminates the effects of the fluctuations in binary stereo labels due to noise in the stereo, illumination changes of the environment from different views, errors in local pose, etc.

A. *Logistic Regression*

The online learner was chosen to be a log-linear module in order to provide lightweight computation for the training at each cycle of video processing. The extracted features X are written to a class-balanced ring buffer, so that samples from previous frames can be reused in the case that there aren't a sufficient number of labeled samples from current frame.

The labels are continuous values indicating the probability of a sample belonging to one of the three classes: occluded, traversable, blocked. The classifier is a logistic regression module and the loss function that is minimized for learning is the Kullback-Liebler divergence or relative entropy (See Figure 4).

$$D_{KL}(P||Q) = \sum_x p(x) log \frac{p(x)}{q(x)}$$

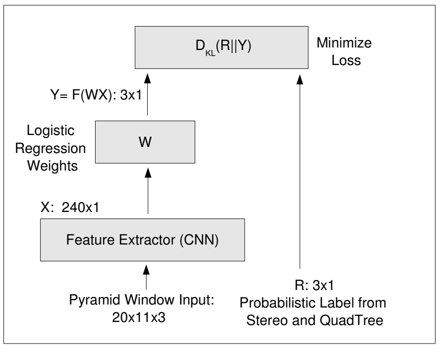
Loss : Kullback- Liebler divergence between the two distributions.

$$Loss = \sum_i r_i log r_i - \sum_i y_i log y_i$$

where

r_i : Desired probability that sample belongs to class i.

y_i : Calculated probability that sample belongs to class i.



$D_{KL}(R||Y)$
Minimize Loss
Y= F(WX): 3x1
Logistic Regression Weights
W
X: 240x1
Feature Extractor (CNN)
Pyramid Window Input: 20x11x3
R: 3x1 Probabilistic Label from Stereo and QuadTree

Fig. 4. Online Learning Architecture.

The outputs are calculated as follows,

$$y_i = \frac{exp(w_k x)}{\sum_k^K exp(w_k x)}$$

The gradient of the $Loss$ with respect to the weights and the update rule,

$$\frac{\partial Loss}{\partial w_j} = \left(\sum_i^K r_i(\delta_{ij} - y_i) \right) \vec{x}$$

$$\Delta w_j = -\eta \left(\sum_i^K r_i(\delta_{ij} - y_i) \right) \vec{x} \quad \delta_{ij} = \begin{cases} 1 & \text{if i=j} \\ 0 & \text{otherwise} \end{cases}$$

A crucial hyper-parameter in online learning is the learning rate, i.e. the size of the update step per sample. A well known drawback of high learning rate is over fitting or loss of generalization due to overly quick adaptation to recently seen samples. This causes a memory loss; i.e., the learned notion of the environments disappear after a while and the robot performs poorly on terrain it has forgotten. One solution is to choose a low η, but this has the disadvantage of lowering the adaptability to new environments. Thus there is a trade-off between responsiveness vs generalization. Figure 6 shows a comparison of classification performance for low and high learning rates. With high η the performance of the classifier for far range is even lower than the not using learning at all.

VII. RESULTS

A. *Offline Error Assesment*

The performance of the long range detection system can be easily illustrated qualitatively, as in Figure 9, where the labels from the stereo and the outputs of the network are projected on to the image space. On the other hand, direct quantitative error assessments during the course of the robot are not feasible. This requires measuring every single pyramid window's real distance from the robot and checking the classifier's output and whether there's an object there. Therefore, we take offline measurements of LROD's performance over the logs after the robots's run.

One way of measuring classification performance is to collect the stereo labels for the entire course of the robot in a map. When the logs are reviewed, this will provide the true class for the windows that don't have stereo labels at hand in a particular frame. Therefore, the classifier's outputs for unlabeled windows can be compared against these labels. Figure 5 illustrates such a collection of stereo labels over the course shown on the left. The graph in Figure 6 shows a comparison of the classification error for different configurations averaged from a test set of logs.

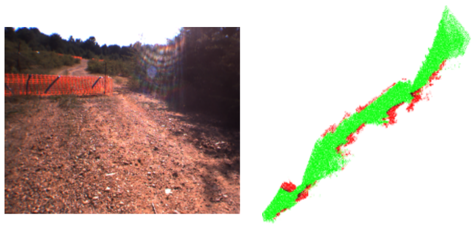


Fig. 5. Right image shows the collection of stereo labels, which provide an answer key for offline testing for the course seen in the image on the left. The robot traverses the soil path for about 100 meters

B. Ground truth error

As mentioned earlier the long range obstacle detection system learns to detect the traversability of the footlines in the robot's range of view. In order to have a quantitative measure of the systems performance human operator sparsely labels several frames from each file in a collection of logs, by tracing obstacle foot lines in the images.

Given the ground truth and long range vision module images, at each direction from the robot's view point, the closest obstacles are compared along each column of both images as highlighted in Figure 7(c).

The possible scenarios are:

- matched: both ground truth and the long range system found an obstacle in the column, the reported error is the distance between those two obstacles
- fake: only the long range system found an obstacle, the error is the maximum distance along the column
- missed: only ground truth found an obstacle, the error is the maximum distance along the column.

Two error metrics are employed, image space and real space. In image space, the distance in pixels of the groundtruth and the network outputs projected onto image space. In local world coordinates, the row and columns are first converted into distance from the robot, in meters, then the resulting distance is the absolute difference of the logs of each distance. The log function is used to normalize the distance errors, to avoid penalizing big errors in meters since here the goal is to measure the performance of an image space classifier. For both cases, the errors are reported.

- ϱ_{fm} and $\varrho_{total-fm}$: error ratio of fake and missed obstacles
- ϱ_a and $\varrho_{total-a}$: error ratio of matched obstacles
- ϱ_{fma} and $\varrho_{total-fma}$: error ratio of fake, missed and matched obstacles

Let d(row1, row2, i) be the function that given a column i in the input image, returns the distance measure between row1 and row2 of this column. When computing the image space measure, d is defined as:

$$d_{img}(row1, row2, i) = |row1 - row2|$$

When computing the real space measure, d is defined as:

$$d_{real}(row1, row2, i) = |log(real(i, row1)) - log(real(i, row2))|$$

The distance error of fake and missed obstacles and of matched obstacles are computed as follow:

$$\varepsilon_{fm}(i) = \begin{cases} d(min, max, i) & \text{if obstacle is fake or missed} \\ 0 & \text{otherwise} \end{cases}$$

$$\varepsilon_a(i) = \begin{cases} d(gt\_row, net\_row, i) & \text{if obstacle is matched} \\ 0 & \text{otherwise} \end{cases}$$

where min and max are the range in which we compare the network with the groundtruth. The error ratio is the distance error over the maximum possible distance error for a frame:

$$\varrho_{fm} = \frac{\sum_{i=1}^{n} \varepsilon_{fm}(i)}{\sum_{i=1}^{n} d(min, max, i)}$$

$$\varrho_a = \frac{\sum_{i=1}^{n} \varepsilon_a(i)}{\sum_{i=1}^{n} d(min, max, i) * matched}$$

where matched is 1 if obstacles match, 0 otherwise. The combined error ratio of fake, missed and matched is a score where fake and missed errors have more weight than distance errors. Indeed fake and missed errors are worse than distance errors.

$$\varrho_{fma} = \frac{\sum_{i=1}^{n} \varepsilon_{fm}(i) + \varepsilon_a(i)}{\sum_{i=1}^{n} d(min, max, i) * (1 + matched)}$$

Finally, the overall error ratios over m frames, which indicate the overall error of fake and missed, matched or both, is:

$$\varrho_{total-\{fma,fm,a\}} = \sum_{j=1}^{m} \varrho_{\{fma,fm,a\}}$$

One can interpret the ground truth comparison visually, with Figure 7(c) where big overlayed stripes of red or pink show the fake and missed obstacles and blue and yellow lines show matched obstacle. Or one can use the error ratios, which are good to compare different systems over the whole set of groundtruth frames.

Table II reports the accumulative ground truth test error from 16 different logs files and a total of 70-75 labelled images. The ratio of missed and fake foot lines from the closest object detected falls from 52.7% to 39.9%. This indicates that there is a significant improvement from the offline trained system to online learning system. And learning with soft-labels outperforms using binary labels clearly.

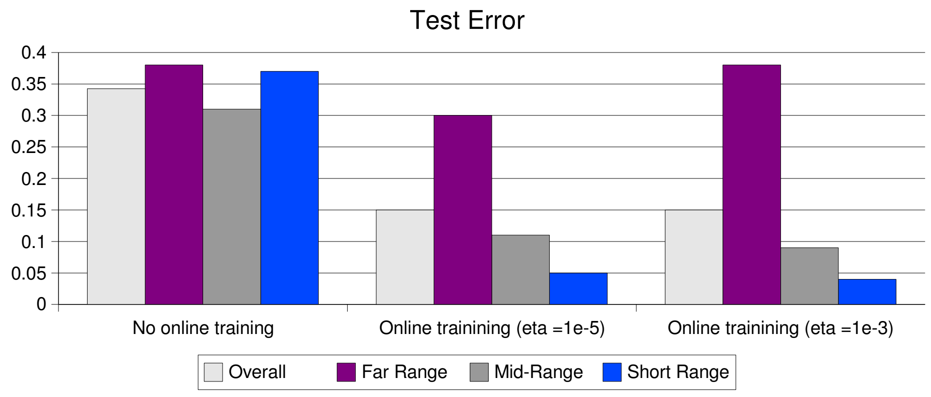


Fig. 6. Accumulated offline test error for different ranges over 9 log files from different terrains. On the left, the classification error without any online training, in the middle online training with low learning rate and on the right online training with high η are shown. When η is high, the error for far bands is even higher than the untrained system, which indicates the generalization is lost.
The distances for the bands are as follows. Far Range [30m - 14.9m], Mid-Range [14.9m - 5.3m], Short Range [5.3m -2.6m]
See section VII-A for the test setting.

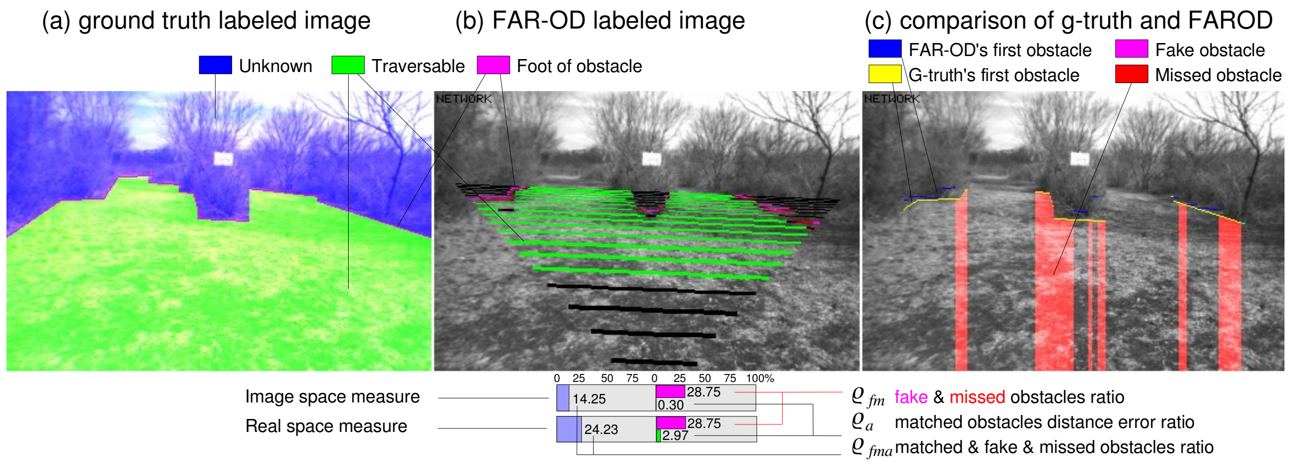


Fig. 7. Ground truth comparison images. Image (a) results from human labeling. Image (b) results from FAR-OD labeling. Image (c) highlights fake and missed errors in red and pink, and shows what obstacles were found by ground truth (yellow) and FAR-OD (blue).

TABLE II
GROUND TRUTH TEST ERROR

	Log Real Missed and Fake (ϱ_{fm})	Log Real Distance (ϱ_a)	Log Real Error Score (ϱ_{fma})	Image Space Missed and Fake (ϱ_{fm})	Image Space Distance (ϱ_a)	Image Space Error Score (ϱ_{fma})
No Online Learning	52.7	0.44	**26.0**	52.5	7,76	**44.6**
Binary Labels	46.9	0.54	**23.3**	46.9	8.05	**40.0**
Soft Labels	39.9	0.57	**19.8**	39.8	9.09	**34.4**

VIII. CONCLUSIONS AND FUTURE WORKS

A long traversability detection system with a range up to 30m is presented. An offline trained feature extractor, which represents a initial notion of the world, is combined with an

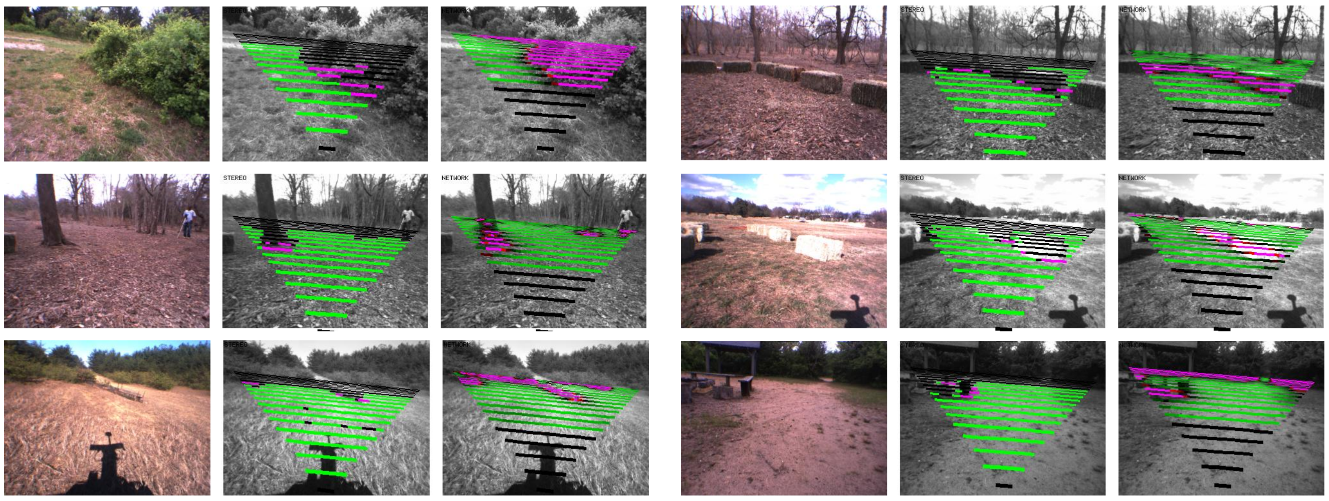


Fig. 8. Examples of desirable classification performance. The left frame shows the input image; the center frame shows the stereo labels that are used to train the classifier; the right frame shows the traversability labels returned by the classifier. Pink is obstacle, green is traversable, and black is unknown. Note that stereo labels are generally sparse and have a maximum range of 12 meters (the last ground lines are always black), whereas the classifier outputs are smooth and consistent and extend to 30 meters. In these examples, obstacles, paths, and traversable are accurately seen far beyond stereo range.

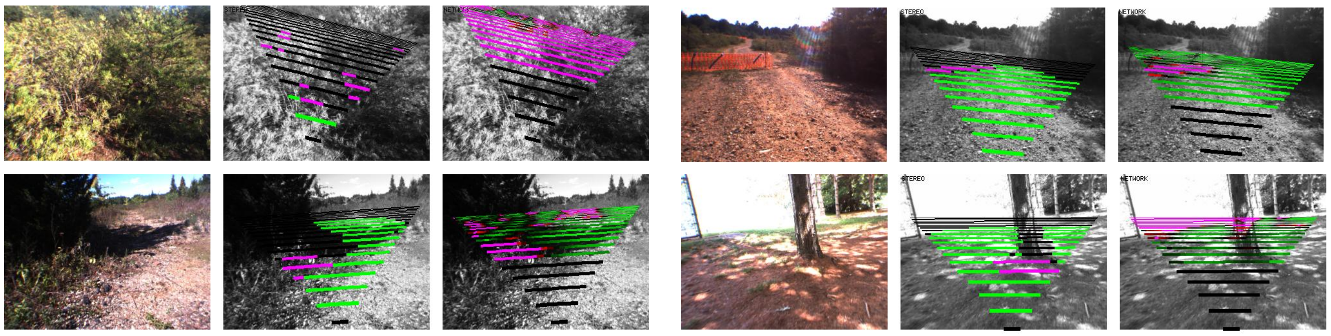


Fig. 9. Examples of poor classification performance. The above examples demonstrate failure modes in the system. If the ground plane estimate is faulty (top left), classification is very difficult. Strong shadows or other phenomena can cause inconsistent, difficult to explain classifier behavior (bottom left). Extreme lighting changes and sun glare often cause false obstacles (bottom right).

online classifier trained these features from a scale-invariant image pyramid. The system gives 85% overall classification accuracy for offline tests over the logs.

One immediate future direction is to relax the single plane fit assumption which would allow the robots perform better in uneven or hilly surfaces. Another one is to embed visual SLAM to the current system. Also an open problem is the active learning of the input samples to preserve online learning over time for long courses.

References

[1] http://www.darpa.mil/ipto/Programs/lagr/vision.htm.

[2] Edward H. Adelson, C. H. Anderson, J. R. Bergen, Peter J. Burt, and J. M. Ogden. Pyramid methods in image processing. *RCA Engineer*, 29(6), 1984.

[3] 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.

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

[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] D Kim, J Sun, S M Oh, J M Rehg, and A F Bobick. Traversibility classification using unsupervised on-line visual learning for outdoor robot navigation. In *Proc. of Int'l Conf. on Robotics and Automation (ICRA)*. IEEE, May 2006.

[7] Y. LeCun and Y. Bengio. Convolutional networks for images, speech, and time-series, 1995.

[8] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[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] David G. Lowe. Object recognition from local scale-invariant features. In *Proc. of the International Conference on Computer Vision ICCV, Corfu*, pages 1150–1157, 1999.

[12] 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.

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

[14] 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.

[15] 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.

[16] Iwan Ulrich and Illah 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.

[17] 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.