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 off-road robot navigation is proposed. The system is purely vision-based and is particularly designed for predicting traversability in unknown or rapidly changing environments. It uses self-supervised learning to quickly adapt to novel 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. The system is developed on the LAGR mobile robot platform and the performance is evaluated using multiple metrics, including ground truth.**

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 efficient goal-driven planning and driving. However, depending on image resolution and processor speeds, stereo algorithms are generally accurate only up to 10 to 12 meters, whereas in open land, camera images contain information far beyond that. On the other hand, the diversity of the terrain and the lighting conditions of outdoor environments make it infeasible to employ a database of obstacle templates or features, or other forms of predefined description collections, which necessitates the use of machine learning techniques. This work focuses on conveying the short range knowledge of the environment to long range vision via self-supervised near-to-far learning.

The learning architecture comprises two parts, a feature extraction module that is trained offline, and an online learning module that allows 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 [14] by Pomerlau, MANIAC [7] by Jochem et al., and DAVE [10] 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 [13], [6]. These systems are trained offline using hand-labeled data which requires significant human effort. Moreover, offline training limits the scope of the robot's expertise to environments seen during training.

To overcome these limitations, navigation systems that are capable of learning traversability labels directly from the environment via *self-supervision* are developed: a reliable sensor provides traversability information learned by a classifier that operates on data from another, less reliable sensor. Not only is the burden of hand-labeling relieved, but the system also becomes flexible to new environments. 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 [4]. In a slightly more complicated approach by Thrun et al.; previous views of the road surface are computed using reverse optical flow, and then road appearance templates are learned for several target distances [11]. Stavens and Thrun used self-supervised learning to train a terrain roughness predictor [16]. An online probabilistic model was trained on satellite imagery and ladar sensor data for the Spinner vehicle's navigation system [15]. 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 [18]. A system that trains a pixel-level classifier using stereo-derived traversability labels is presented by Ulrich [17]. Recently Kim et al. [8] proposed an autonomous off-road navigation system that estimates traversability in an unstructured, unknown outdoor environment.

The proposed system incorporates feature extraction and label propagation into a self-supervised online learning framework that is designed for maximum flexibility and adaptability in changing, off-road environments.

III. OVERVIEW OF THE SYSTEM

As mentioned previously, the proposed long range obstacle detection system (LROD) comprises two parts: a feature extractor that is used to transform image patches to a lower dimensional and more discriminative representation, 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. 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 performs a series of computations, including pre-

TABLE I

OVERVIEW OF PRINCIPLE PROCESSING STEPS IN THE VISION SYSTEM.

Processing Step	**Processing Time**
Pre-processing	
Image rectification and point cloud extraction	45 ms
Ground plane estimation	35 ms
Conversion to YUV and normalization	40 ms
Horizon leveled, distance-normalized pyramid	10 ms
Labeling	
Stereo labeling of windows in pyramid	20 ms
Feature Extraction	
Feature extraction (convolutional neural network)	85 ms
Label Propagation	
Query quad-tree for matching windows	10 ms
Label query results with probabilistic labels	0 ms
Insert feature vectors into quad-tree	0 ms
Add labeled samples to ring buffer	1 ms
Online Training and Classification	
Train logistic regression on ring buffer contents	40 ms
Classify all windows in pyramid	5 ms
Total	291 ms

processing, feature extraction, training, and classification steps. The steps in one full processing cycle are listed in order in Table I, along with the average processing time for each step. The LAGR Robot has four dual-core processors, two of which are dedicated purely to visual computations. This allows a frame rate of 2-3 Hz and thus real time processing in accordance with the other parts of the system. Section IV discusses the image pre-processing and the feature extraction, and Section VI describes the online label propagation and training strategies. The 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, the visual data in these images must be transformed into discrete windows of information and each window 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 then obtain a point cloud in RCD (row, column, disparity) space: $P = (r_1, c_1, d_1), (r_2, c_2, d_2), ..., (r_n, c_n, d_n)$ using the Triclops SDK [2]. From this 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 by analyzing the principle components of 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, correspondence 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:

$$x = \frac{x}{\sum_{y \in I^W, k \in K} yk + 1}$$

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 [3]), and more recently have been used for scale-invariant object recognition (see [12]). 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 *foot line* on the ground that is at a fixed distance from the robot. There are 24 foot lines 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. The points are collected in bins that correspond to the real world coordinates of windows in the pyramid. Simple heuristics are used to decide whether each window's bin corresponds to a traversable (*ground*) or non-traversable (*obstacle*) area, based on the groundplane distance of the points in the bin and their variance. The window could also be labeled as *blocked*, if there is a nearby obstacle that occludes that window. Windows in the pyramid are thus labeled (ground, obstacle, or blocked) according to the RCD points at the foot line of the window (see Figure 2).

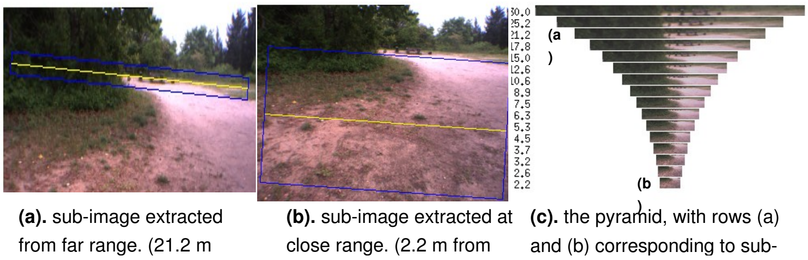

Fig. 1. Sub-images (shown as blue rectangles) are extracted according to imaginary lines on the ground (shown in yellow) which are computed using the estimated ground plane. *(a)* Extraction around a foot line that is 21m away from the vehicle. *(b)* Extraction around a foot line that is 1.1m away from 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 foot line is on open ground, so label = ground; *center*: the base of the object is on the foot line, so label = obstacle; *right*: the foot line is blocked by a nearby object, so label = blocked.

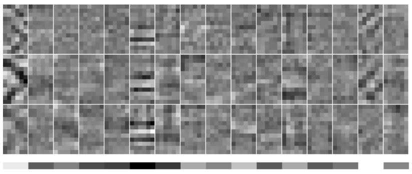

Fig. 3. The kernels learned by the feature extractor using offline training show a sensitivity to horizontal boundary lines.

V. FEATURE EXTRACTION

A convolutional neural network (CNN) [9] contains local receptive fields that are trained to extract local features and patterns. This architecture makes the CNN naturally shift and scale invariant, and therefore ideal for learning discriminative visual features. The 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. For the purposes of training, a final fully-connected layer exists with 3 outputs. After the network is trained, the fully connected layer is removed and the online output of the network is a 240 dimension feature vector. The filters (see Figure 3) show that the network is very responsive to horizontal structures, such as obstacle feet and other visual boundaries.

The CNN is the only component of our system that is trained offline rather than online. The offline data set consists of 600,000 samples taken randomly from 130 diverse logfiles. Image preprocessing and stereo labeling of these samples was identical to the online process described in the previous section. The stereo labels were "smoothed" using the propagation scheme described in Section VI. The CNN had a final error rate of 20.59%

VI. ONLINE LEARNING

Throughout this paper, online learning is used to refer to the near-to-far learning of long-range traversability as the robot traverses a course. 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. As the robot proceeds through the environment, it collects features and the associated stereo labels in this map. From this collection, soft-labels for the pyramid windows are calculated as the ratios of the labels accumulated in the quadtree cells corresponding to their real world locations. 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 on each frame of video processing. The logistic regression module has three outputs corresponding to the probabilities of a sample belonging to each one of the three categories: occluded, traversable, and blocked. The loss function that is minimized for learning is the Kullback-Liebler divergence or relative entropy (See Figure 4).

$$Loss = D_{KL}(P||Q) = \sum_{i=1}^{K} p_i log p_i - \sum_{i=1}^{K} p_i log q_i$$

where p_i is the probability that the sample belongs to class i calculated from the stereo labels collected in the quadtree. q_i is the classifier's output for the probability that the sample belongs to class i.

$$q_i = \frac{exp(\mathbf{w_i x})}{\sum_{k=1}^{K} exp(\mathbf{w_k x})}$$

where $\mathbf{w}$ are the parameters of the classifier, and $\mathbf{x}$ is the sample's feature vector.

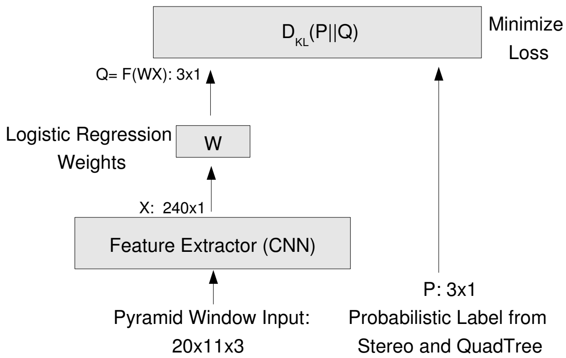

Fig. 4. Online Learning Architecture.

The update rule is,

$$\Delta \mathbf{w_j} = -\eta \frac{\partial Loss}{\partial \mathbf{w_j}} = -\eta \left(\sum_{i=1}^{K} p_i(\delta_{ij} - q_i) \right) \mathbf{x}$$

$$\delta_{ij} = \begin{cases} 1 & \text{if i=j} \\ 0 & \text{otherwise} \end{cases}$$

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 using high learning rate is overfitting or loss of generalization due to overly quick adaptation to recently seen samples i.e. knowledge of an environment fades rapidly and the robot performs poorly on terrain it has forgotten. One solution is to choose a low learning rate (η), but this has the disadvantage of reducing the responsiveness to new environments. Thus, there is a trade-off between responsiveness and 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 worse than the case when no learning is used at all.

VII. RESULTS

A. Offline Error Assessment

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. However, direct quantitative error assessments as the robot drives are not feasible. Therefore, we take offline measurements of the performance over the logs after the robot'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 provides the true labels 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.

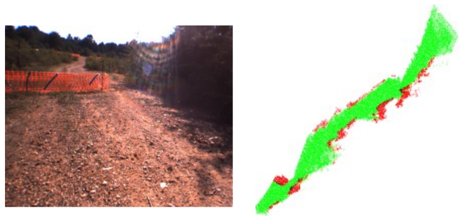

Fig. 5. Right image shows the collection of stereo labels, which provide an answer key for offline testing for the course seen in the image on the left. The robot traverses the soil path for about 100 meters.

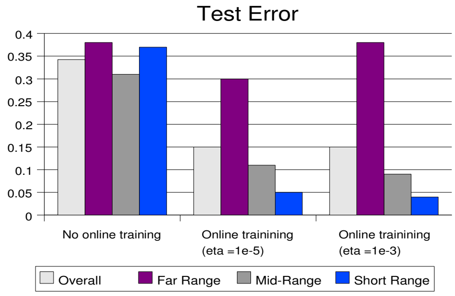

Fig. 6. Offline test error for different ranges over 9 log files from different terrains. From left to right, the classification error without any online training, online training with low learning rate (η), and with high η are shown. In the last case, the error for far bands is even higher than the first, indicating that the generalization is lost. The distances for the bands: 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.

B. Ground truth error

In this offline test setting, a human operator labels several frames from each file in a collection of logs, by tracing obstacle foot lines. Given the ground truth labels and long range vision module outputs, the closest obstacles are compared as illustrated in Figure 7(c). The possible comparisons 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.

The ground truth error is measured in both image space and real space. Let $d(row1, row2, i)$ be the function which, given a column i in the input image, returns the distance between $row1$ and $row2$ of this column. In image space, d is calculated as the pixel distance of the ground truth foot lines and the network outputs projected onto image space.

$$d_I(row1, row2, i) = |row1 - row2|$$

On the other hand, when computing the real space distance, the row and columns are first converted into real world

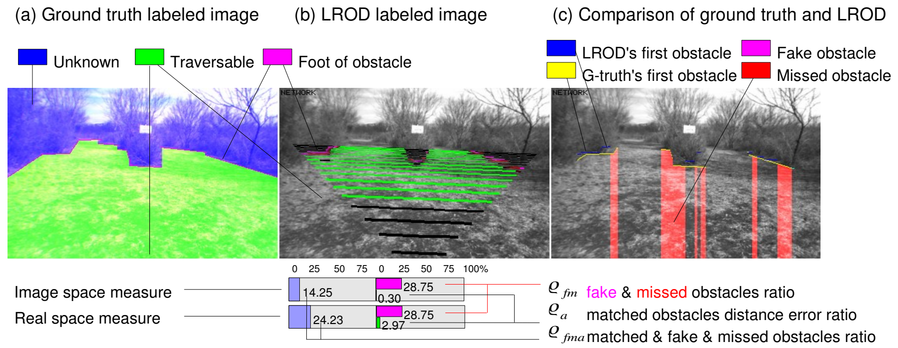

Fig. 7. Ground truth comparison images. Image (a) results from human labeling. Image (b) results from LROD labeling. Image (c) highlights fake and missed errors in red and pink, and shows what obstacles were found by ground truth (yellow) and LROD (blue).

distances and d is calculated as,

$$d_R(row1, row2, i) = |log(real(i, row1)) - log(real(i, row2))|$$

We defined the following error metrics for each case,

- ϱ_{fm}: error ratio of fake and missed obstacles.

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

- ϱ_a: error ratio of matched obstacles.

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

- ϱ_{fma}: the combined error ratio of fake, missed and matched obstacles.

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

where *matched* is 1 if obstacles match, 0 otherwise, and, *min* and *max* are the limits of the range in the space where the error metric is calculated.

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

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

Finally, the overall error ratios over m frames are defined as,

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

One can interpret the ground truth comparison visually, as in Figure 7(c) where big overlayed stripes of red or pink show the fake and missed obstacles, and blue and yellow lines show matched obstacles. Or one can use the error ratios, which are good to compare different systems over the whole set of ground truth frames. Table II reports the ground truth test error from 16 different log files and a total of 70-75 labeled 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. We can also see that learning with soft probabilistic labels clearly outperforms the use of binary labels.

This work presents and evaluates the performance of the long range traversability detection module only. For the tests of the overall system in unstructured environments see [5].

VIII. CONCLUSIONS AND FUTURE WORKS

A self-supervised terrain traversability classification system with a range up to 30m is presented. An offline trained feature extractor, which represents an initial notion of the world, is combined with an online classifier trained on short range stereo information. The system gives 85% overall classification accuracy in offline tests over the logfiles.

One immediate future goal is to relax the single ground plane assumption which would improve the robot's performance in uneven or hilly surfaces. Another direction is employing active learning in order to preserve generalization and to avoid memory loss over time for long courses.

REFERENCES

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

[2] http://www.ptgrey.com/products/triclopsSDK/index.asp.

[3] E. Adelson, C. H. Anderson, J. R. Bergen, P. Burt, and J. Ogden. Pyramid methods in image processing. *RCA Engineer*, 29(6), 1984.

[4] 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.

[5] R. Hadsell, A. Erkan, P. Sermanet, J. Ben, K. Kavukcuoglu, U. Muller, and Y. LeCun. A multi-range vision strategy for autonomous offroad navigation. In *IASTED International Conference on Robotics and Applications (RA)*, 2007.

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

[7] T. Jochem, D. Pomerleau, and C. 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.

[8] 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, 2006.

[9] Y. LeCun and Y. Bengio. Convolutional networks for images, speech, and time-series. In *The Handbook of Brain Theory and Neural Networks*. MIT Press., 1995.

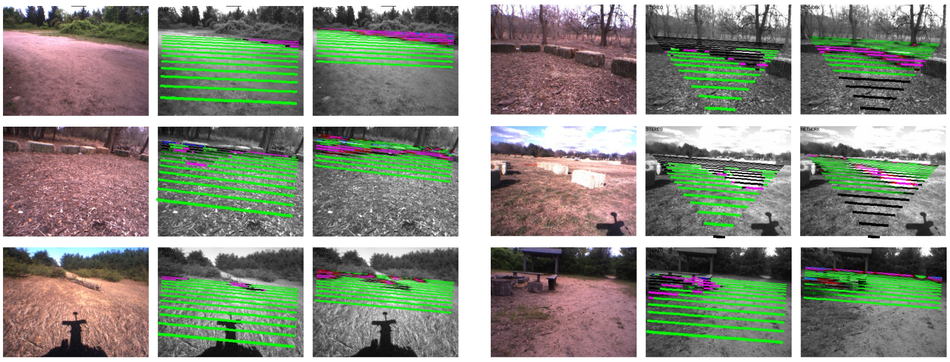

Fig. 8. Examples of desirable classification performance. The left frame shows the input image; the center frame shows the stereo labels that are used to train the classifier; the right frame shows the traversability labels returned by the classifier. Pink is obstacle, green is traversable, and black 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.

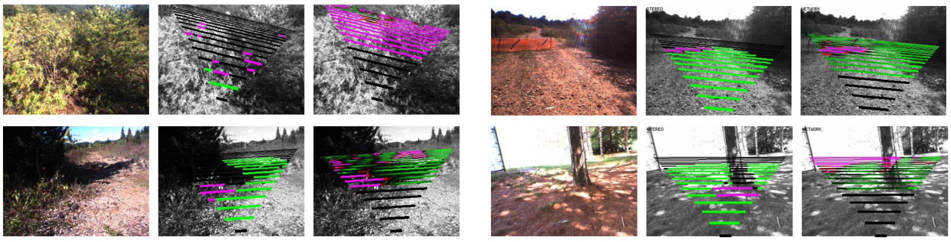

Fig. 9. Examples of poor classification performance. The above examples demonstrate failure modes in the system. If the ground plane estimate is inaccurate (top left), classification is very difficult. Strong shadows or other phenomena can cause inconsistent, difficult to explain classifier behavior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

TABLE II
GROUND TRUTH TEST ERROR

	Log Real Missed and Fake ($\varrho_{total-fm}$)	Log Real Distance ($\varrho_{total-a}$)	Log Real Error Score ($\varrho_{total-fma}$)	Image Space Missed and Fake ($\varrho_{total-fm}$)	Image Space Distance ($\varrho_{total-a}$)	Image Space Error Score ($\varrho_{total-fma}$)
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**

[10] 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.

[11] 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.

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

[13] 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.

[14] D. Pomerlau. Knowledge based training of artificial neural networks for autonomous driving. *Robot Learning*, 1993.

[15] 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.

[16] 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.

[17] 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.

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