Deep Belief Net Learning in a Long-Range Vision System for Autonomous Off-Road Driving

Raia Hadsell[1] Ayse Erkan[1] Pierre Sermanet[1,2] Marco Scoffier[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*— We present a learning process for long-range vision that is able to accurately classify complex terrain at distances up to the horizon, thus allowing high-level strategic planning. A deep belief network is trained to extract informative and meaningful features from an input image, and the features are used to train a realtime classifier to predict traversability. A hyperbolic polar coordinate map is used to accumulate the terrain predictions of the classifier. The process was developed and tested on the LAGR mobile robot.**

I. INTRODUCTION

Humans navigate effortlessly through most outdoor environments, detecting and planning around distant obstacles even in new, never-seen terrain. Shadows, hills, groundcover variation - none of these affect our ability to make strategic planning decisions based purely on visual information. These tasks, however, are extremely challenging for a vision-based mobile robot. Current robotics research has only begun to develop vision-based systems that can navigate through offroad environments. However, existing approaches to this problem often rely on stereo algorithms, which produce short-range, sparse, and noisy costmaps that are inadequate for long-range strategic navigation. Stereo algorithms are limited by image resolution, and often fail when confronted by repeating or smooth patterns, such as tall grass, dry scrub, or smooth pavement. Current research has focused on increasing the range of vision by classifying terrain in the far field according to the color of nearby ground and obstacles. This type of near-to-far color-based classification is quite limited, however. Although it gives a larger range of vision, the classifier has low accuracy and can easily be fooled by shadows, monochromatic terrain, and complex obstacles or ground types.

The long-range vision system that we propose uses self-supervised learning to train a classifier in realtime. To successfully learn a complex environment, the classifier must be trained with *discriminative features extracted from large image patches*, and the features must be labeled with *visually consistent* categories. For the classifier to successfully generalize from near to far field, the training samples must be *normalized with respect to scale and distance*. The first criterion, training with large image patches, is crucial for true recognition of obstacles, paths, groundtypes, and other natural features. Color histograms or texture gradients cannot replace the contextual information in actual image patches. The second criterion, visually consistent labeling, is equally important for successful learning. The classifier is trained using labels generated by stereo processing. If the label categories are inconsistent or extremely noisy, the learning will fail. Therefore, our stereo-based supervisor module uses 5 categories that are visually distinct: *super-ground*, *ground*, *footline*, *obstacle*, and *super-obstacle*. The supervisor module is designed to limit incorrect or misaligned labels as much as possible. The third criterion, normalization with respect to size and distance, is necessary for good generalization. We normalize the image by constructing a horizon-leveled input pyramid in which similar obstacles are similar heights, regardless of their distance from the camera.

The long-range vision classifier was developed and tested as part of a full navigation system. The outputs from the classifier populate a hyperbolic polar coordinate costmap, and planning algorithms are run on the map to decide trajectories and wheel commands at each step. The long-range classifier has been tested independently, and experiments have also been conducted to assess the impact of the long-range vision on the navigation performance of the robot. We show that on multiple test courses, the long-range vision yields driving that is smoother and more far-sighted. We also show the performance of the classifier in several diverse environments.

A. The LAGR Platform

The vision system described here was developed on the LAGR platform (see Fig. 1). LAGR (Learning Applied to Ground Robots) is a DARPA program [1] in which participants must develop learning and vision algorithms for an outdoor, offroad autonomous vehicle. The LAGR robot has 4 onboard computers, 2 stereo camera pairs, a GPS receiver, and an IMU (inertia measurement unit). It has a maximum speed of 1.2 meters/second.

II. PREVIOUS WORK

Many methods for vision-based navigation rely on stereo-based obstacle detection [9], [11], [3]. A stereo algorithm finds pixel disparities between two aligned images, producing a 3d point cloud. By applying heuristics to the statistics of groups of points, obstacles and ground are identified.

Fig. 1. The LAGR mobile robotic vehicle, developed by Carnegie Mellon University's National Robotics Engineering Center. Its sensors consist of 2 stereo camera pairs, a GPS receiver, and a front bumper.

However, the resulting costmaps are often sparse and short-range.

Recent approaches to vision-based navigation use learning algorithms to map traversability information to color histograms or geometric (point cloud) data. This is especially useful for road-following vehicles [2], [12], [8]; the ground immediately in front of the vehicle is assumed to be traversable, and the rest of the image is then filtered to find similarly colored or textured pixels. Although this approach helped to win the 2005 DARPA Grand Challenge, its utility is limited by the inherent fragility of color-based methods.

Other, non-vision-based systems have used the near-to-far learning paradigm to classify distant sensor data based on self-supervision from a reliable, close-range sensor. Stavens and Thrun used self-supervision to train a classifier to predict surface roughness [15]. A self-supervised classifier was trained on satellite imagery and ladar sensor data for the Spinner vehicle's navigation system [14]. An online self-supervised classifier for a ladar-based navigation system was trained to predict load-bearing surfaces in the presence of vegetation [16].

Predictably, the greatest similarity to our proposed method can be found in the research of other LAGR participants. Since the LAGR program specifically focused on learning and vision algorithms that could be applied in new, never-seen terrain, using near-to-far self-supervised learning was a natural choice [4], [6], [10].

Our approach differs from the aforementioned research because it uses a deep belief network to extract features from large image patches, then trains a classifier to learn to discriminate these feature vectors into 5 classes. We build a distance-normalized, horizon-leveled image pyramid to deal with the limitations of generalization from near to far field.

III. Overview of Long Range Vision

The long-range vision system is a self-supervised, realtime learning process (see Fig. 2). The only input is a pair of stereo-aligned images, and the output is a set set of points in vehicle-relative coordinates, each one labeled with a vector of 5 energies, corresponding to 5 possible categories. The points and their energy vectors are used to populate a large

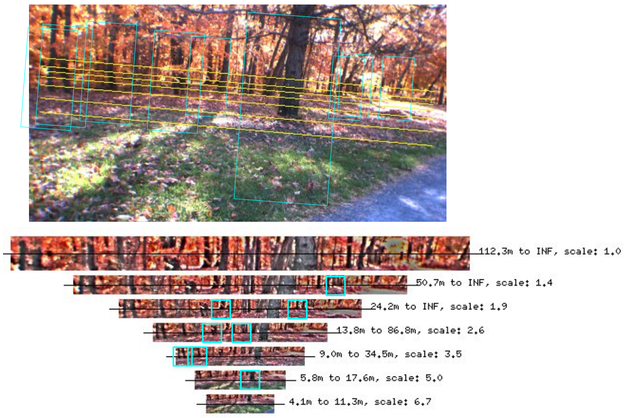

Fig. 3. The input image at top has been methodically cropped and leveled and subsampled to yield each pyramid row seen at the bottom. The bounding boxes demonstrate the effectiveness of the normalization: trees that are different scales in the input image are similarly scaled in the pyramid.

polar coordinate map. Path planning algorithms are run on the polar map, which in turn produce driving commands. The primary components of the learning process are briefly listed.

Pre-processing and Normalization. Pre-processing is done to level the horizon and to normalize the height of objects independently of distance.

Stereo Supervisor Module. The stereo supervisor assigns class labels to close-range input windows.

Feature extraction. Features are extracted from input windows using stacked layers of convolutional features. This deep belief network is trained offline.

Training and Classification. The classifier is trained on every frame for fast adaptability.

IV. Horizon-Leveling and Normalization

We are strongly motivated to use large image patches (large enough to fully capture a natural element such as a tree or path) because larger context and greater information yields better learning and recognition. However, the problem of generalizing from nearby objects or groundtypes to far objects is daunting, since apparent size scales inversely with distance: Angular size $\propto \frac{1}{\text{Distance}}$. Our solution is to create a normalized pyramid of 7 sub-images which are extracted at geometrically progressing distances from the camera. Each sub-image is subsampled according to that estimated distance, yielding a set of images in which similar objects have a similar pixel height, regardless of their distance from the vehicle (see Fig. 3). The closest pyramid row has a target range from 4 to 11 meters away and is subsampled with a scaling factor of 6.7. The furthest pyramid row has a range from 112 meters to ∞ (beyond the horizon) and has a scaling ratio of 1 (no subsampling).

A bias in the roll of the cameras, plus the natural bumps and grading in the terrain, means that the horizon is generally skewed in the input image. We normalize the

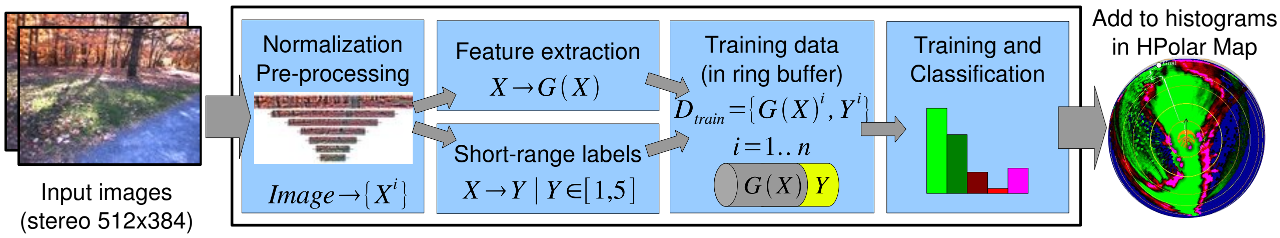

Fig. 2. Diagram of the Long-Range Vision Process. The input to the system is a pair of stereo-aligned images. The images are normalized and features and labels are extracted, then the classifier is trained and immediately used to classify the entire image. The classifier outputs are accumulated in histograms in a hyperbolic polar map according to their (x, y) position.

horizon position in the pyramid by explicitly estimating the location of the horizon. First we estimate the groundplane $p = \{p_0, p_1, p_2, p_3\}$ using a Hough transform on the stereo point cloud, then refine that estimate using a PCA robust refit. Once p is known, the horizon can be leveled: left-y-offset $= \frac{(p_1 * .5w) + p_2 + p_3}{-p_0} - (.5w * \sin\alpha)$, where w is the width of the input image and α is the angle of the skewed horizon.

The input is also converted from RGB to YUV, and the Y (luminance) channel is contrast normalized to alleviate the effects of hard shadow and glare. The contrast normalization performs a smooth neighborhood normalization on each y in Y by normalizing by the linear sum of a smooth 16x16 kernel and a 16x16 neighborhood of Y (centered on y).

V. Feature Learning: Deep Belief Network

Normalized overlapping windows (3@25x12 pixels) from the pyramid rows provide a basis for strong near-to-far learning, but the curse of dimensionality makes it impossible to directly train a classifier on the YUV windows. Feature extraction lowers the dimensionality while increasing the generalization potential of the classifier. There are many ways that feature extraction may be done, from hand-tuned feature lists, to quantizations of the input, to learned features. We prefer to use learned features, because they can capture patterns in the data that are easily missed by a human.

We have experimented in the past with extracting features with radial basis functions and with supervised trained convolutional networks [5], but had only moderate success: the radial basis functions, trained using k-means unsupervised clustering, produced stable feature vectors that were not discriminative enough and lead to weak online learning, and the supervised convolutional network learned filters that were noisy and unclear, which caused unpredictable online learning.

The current approach uses the principles of *deep belief network training* [7], [13]. The basic idea behind deep belief net training is to pre-train each layer independently and sequentially in unsupervised mode using a reconstruction criterion to drive the training. The deep belief net trained for the long-range vision system consists of 3 stacked modules. The first and third modules are convolutional layers, and the second layer is a max-pooling unit. Each convolutional layer can be understood as an *encoder* $F_{enc}(X)$ that creates a set of features from the given input by applying a sequence of convolutional filters. A *decoder* $F_{dec}(Y)$ tries to recreate the input from the feature vector output. The encoder and decoder are trained simultaneously by minimizing the reconstruction error, i.e., minimizing the mean square loss between the input and the encoded and decoded reconstruction:

$$\mathcal{L}(S) = \frac{1}{P} \sum_{i=1}^{P} ||X^i - F_{dec}(F_{enc}(X^i))||_2$$

where S is a dataset with P training samples.

A. Deep Belief Net Architecture and Training

As stated, the network trained for feature extraction in the long-range classifier consists of 3 stacked modules. The first and third are convolutional layers, composed of a set of convolutional filters and a point-wise non-linearity. The function computed for an input layer x and filter f and output feature map z is

$$z_j = \tanh(c_j(\sum_i x_i * f_{ij}) + b_j)$$

where $*$ denotes the convolution operator, i indexes the input layer, j indexes the output feature map, and c_j and b_j are multiplicative and additive constants. The max-pooling layer is used to reduce computational complexity and to pool features, creating translation invariance. The max-pooling operation, for input layer x and output map z, is

$$z_i = \max_{i \in N_i}(x)$$

where N_i is the spatial neighborhood for max-pooling.

The first convolutional layer of the feature extractor has 20 7x6 filters, and 20 feature maps are output from the layer. After max-pooling with a kernel size of 1x4, the pooled feature maps are input to the second convolutional layer, which has 300 6x5 filters and produces 100 feature maps. Each overlapping window in the input has thus been reduced to a 100x1x1 feature vector. The feature maps for a sample input (a row in the normalized pyramid) are shown in Fig. 4.

The feature extractor was trained until convergence using the deep belief net training protocol described above. The training set was composed of images from 150 diverse outdoor settings, comprising 10,000 images in total, with each image further scaled to different resolutions. The convolutional filters are shown in Fig. 5.

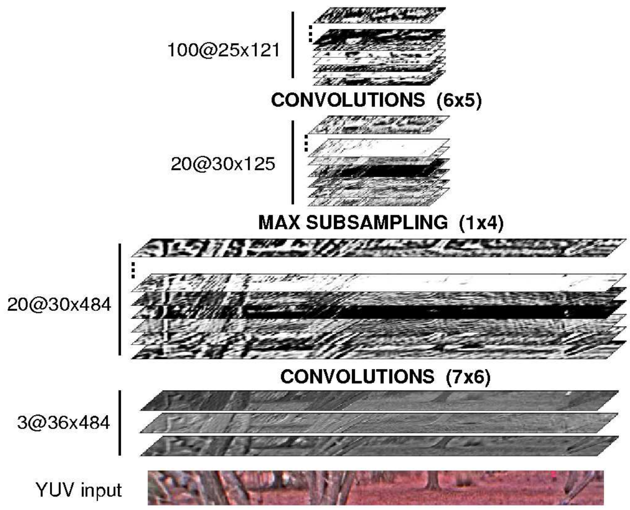

Fig. 4. The feature maps are shown for a sample input. The input to the network is a variable width, variable height layer from the normalized pyramid. The output from the first convolutional layer is a set of 20 feature maps, the output from the max-pooling layer is a set of 20 feature maps with width scaled by a factor of 4 through pooling, the output from the second convolutional layer is a set of 100 feature maps. A single 3x12x25 window in the input corresponds to a single 100x1x1 feature vector.

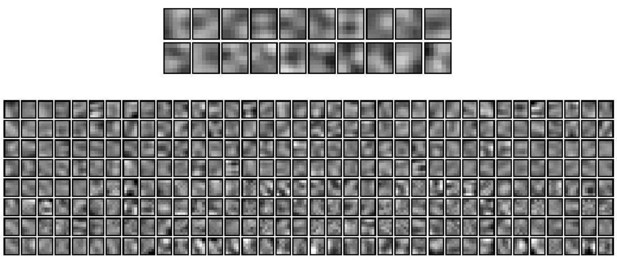

Fig. 5. Trained filters from both layers of the trained feature extractor. *Top:* the first convolutional layer has 20 7x6 filters. *Bottom:* the second convolutional layer has 300 6x5 filters.

VI. Stereo Supervision

The supervision that the long-range classifier receives from the stereo module is critically important. The realtime training can be dramatically altered if the data and labels are changed in small ways, or if the labeling becomes noisy. Therefore, the goal of the supervisor module is to provide data samples and labels that are *visually consistent*, *error-free*, and *well-distributed*. The basic approach begins with a disparity point cloud generated by a stereo algorithm. In the first step, the ground plane is located within the point cloud and the points are separated with a threshold into ground and obstacle point clouds. Second, the obstacle points are projected onto the ground plane to locate the feet of the obstacles. Third, overlapping regions of points are considered and heuristics are used to assign each region to one of five categories. The results from this basic method have several sources of error, however. Since off-road terrain is rarely perfectly flat, areas of traversable ground can stick up above the ground plane and be mis-classified as obstacle. Also, tufts of grass or leaves can look like obstacles when a simple plane distance threshold is used to detect obstacles. These sorts of error are potentially disastrous to the classifier, so two strategies, multi-groundplane estimation and moments filtering, are employed to avoid them.

Multi-Groundplane Estimation The assumption that there is a single perfect ground plane is rarely correct in natural terrain. To relax this assumption, we find multiple planes in each input image and use their combined information to divide the points into ground and obstacle clouds. After the first ground plane is fitted to the point cloud, all points that are within a tight threshold of the plane are removed and a new plane is fit to the remaining points. The process continues until no good plane can be found or a maximum of 4 planes have been fit to the data. The ground planes are fitted by using a Hough transform to get an initial estimate of the plane, then refining the estimate by locating the principle eigenvectors of the points that are close to the initial plane.

Moments Filtering Even multiple ground planes cannot remove all error from the stereo labeling process. Tufts of grass or disparity mismatches (common with repeating textures such as tall grass, dry brush, or fences) can create false obstacles that cause poor driving and training. Therefore, we consider the first and second moments of the plane distances of points and use the statistics to reject false obstacles. We use the following heuristics: if the mean plane distance is not too high (under .5 m) and the variance of the plane distance is very low, then the region is traversable (probably a traversable hillside). Conversely, if the mean plane distance is very low but the variance is higher, then that region is traversable (possibly tall grass). These heuristics are simple, but they reduce errors in the training data effectively.

Visual Categories Most classifiers that attempt to learn terrain traversability are binary; they only learn ground vs. obstacle. However, our supervisor uses 5 categories: *super-ground*, *ground*, *footline*, *obstacle*, and *super-obstacle*. *Super-ground* and *super-obstacle* refer to input windows in which *only* ground or obstacle are seen, and our confidence is very high that these labels are correct. The *ground* and *obstacle* categories are used when other sorts of terrain are seen in the window, or when the confidence is lower that the label is correct. *Footline* is the label for input windows that have the foot of an obstacle centered in the window. Obstacle feet are visually distinctive, and it is important to put these samples in a distinct. category. Fig. 6 and Fig. 7 show examples of the 5 categories.

VII. Realtime Terrain Classification

The long-range classifier trains on and classifies every frame that it receives, so it must be relatively efficient. A separate logistic regression is trained on each of the 5 categories, using a one-against-the-rest training strategy. The loss function that is minimized for learning is the Kullback-Liebler divergence or relative entropy. $Loss = D_{KL}(P||Q) = \sum_{i=1}^{K} p_i log p_i - \sum_{i=1}^{K} p_i log q_i$, where p_i is

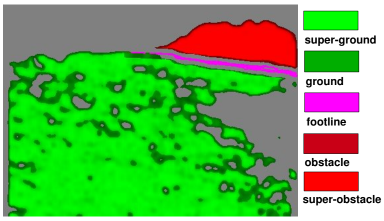

Fig. 6. This shows the 5 labels applied to a full image.

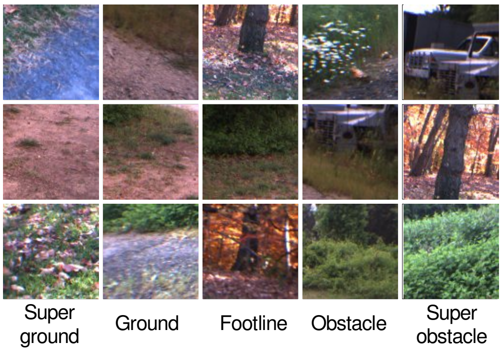

Fig. 7. Examples of the 5 labels. Although the classes include very diverse training examples, there is still benefit to using more than 2 classes. *Super-ground*: only ground is seen in the window; high confidence, *ground*: ground and obstacle may be seen in window; lower confidence, *footline*: obstacle foot is centered in window, *obstacle*: obstacle is seen but does not fill window; lower confidence, and *super-obstacle*: obstacle fills window; high confidence.

the probability that the sample belongs to class i calculated from the stereo supervisor labels. 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. The weights of each regression are updated using stochastic gradient descent, since gradient descent provides strong regularization over successive frames and training iterations. The gradient update is

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

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

VIII. Results

The long-range vision system has been used extensively in the full navigation system built on the LAGR platform. It runs at 1-2 Hz, which is too slow to maintain good close-range obstacle avoidance, so the system architecture runs 2 processes simultaneously: a fast, low-resolution stereo-based obstacle avoidance module and planner run at 8-10 Hz and allow the robot to nimbly avoid obstacles within a 5 meter radius. Another process runs the long-range vision and long-range planner at 1-2 Hz, producing strategic navigation and planning from 5 meters to the goal.

We present experimental results obtained by running the robot on 2 courses with the long-range vision turned on and turned off. With the long-range vision turned off, the robot relies on its fast planning process and can only detect obstacles within 5 meters. Course 1 (see Fig. 8 top and Table 9) is a narrow wooded path that proved very difficult for the robot with long-range vision off, since the dry scrub bordering the path was difficult to see with stereo alone. The robot had to be rescued repeatedly from entanglements off the path. With long-range vision on, the robot saw the scrub and path clearly and drove cleanly down the path to the goal. Course 2 (see Fig. 8 bottom and Table 9) was a long wide path with a clearing to the north that had no outlet - a large natural cul-de-sac. Driving with long-range vision on, the robot saw the long path and drove straight down it to the goal without being tempted by the cul-de-sac. Driving without long-range vision, the robot immediately turned into the cul-de-sac and became stuck in scrub, needing to be manually driven out of the cul-de-sac and restarted in order to reach the goal.

Fig. 10 shows 5 examples of long-range vision in very different terrain. The input image, the stereo labels, and the classifier outputs are shown in each case.

IX. Conclusions

We have described, in detail, an self-supervised learning approach to long-range vision in off-road terrain. The classifier is able to see smoothly and accurately to the horizon, identifying trees, paths, man-made obstacles, and ground at distances far beyond the 10 meters afforded by the stereo supervisor. Complex scenes can be classified by our system, well beyond the capabilities of a color-based approach. The success of the classifier is due to the use of large context-rich image windows as training data, and to the use of a deep belief network for learned feature extraction.

Acknowledgments

The authors wish to thank Larry Jackel, Dan D. Lee, and Martial Hébert for helpful discussions. This work was supported by DARPA under the Learning Applied to Ground Robots program.

References

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

[2] 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.

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

[4] G. Grudic and J. Mulligan. Outdoor path labeling using polynomial mahalanobis distance. In *Proc. of Robotics: Science and Systems (RSS)*, August 2006.

[5] R. Hadsell, P. Sermanet, J. Ben, A. Erkan, J. Han, B. Flepp, U. Muller, and Y. LeCun. Online learning for offroad robots: Using spatial label propagation to learn long-range traversability. In *rss*, 2007.

[6] M. Happold, M. Ollis, and N. Johnson. Enhancing supervised terrain classification with predictive unsupervised learning. In *Proc. of Robotics: Science and Systems (RSS)*, August 2006.

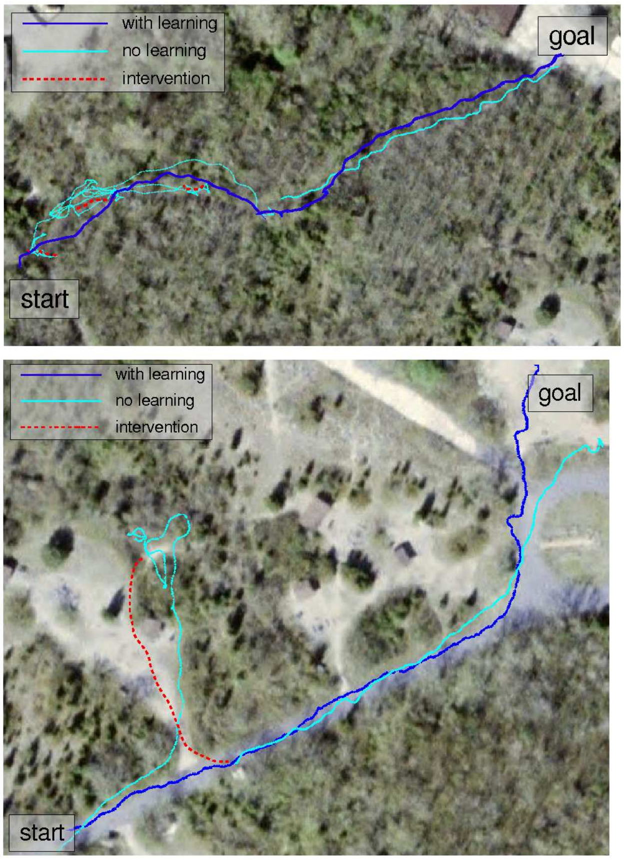

Fig. 8.

Course 1	Total Time	Total Distance	Inter-ventions
No Long-Range	321 sec	271.9 m	3
With Long-Range	155.5 sec	166.8 m	0
Course 2			
No Long-Range	196.1 sec	207.5 m	1
With Long-Range	142.2 sec	165.1 m	0

Fig. 9.

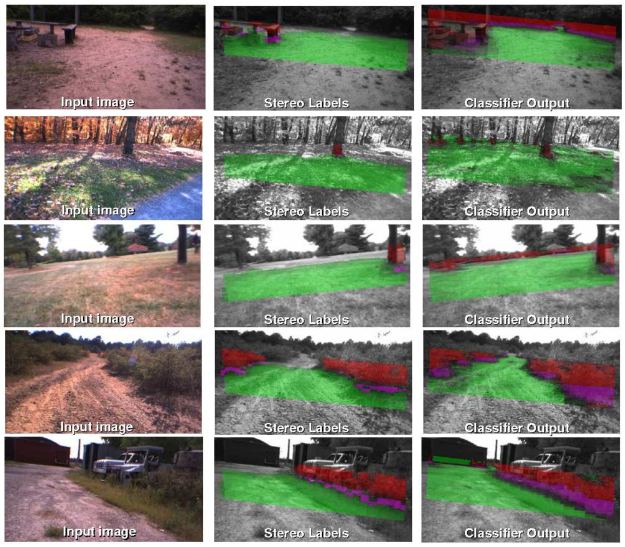

Fig. 10.

[7] G. E. Hinton, S. Osindero, and Y. Teh. A fast learning algorithm for deep belief nets. *Neural Computation*, 18:1527–1554, 2006.

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

[9] 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*, 1998.

[10] 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.

[11] 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.

[12] 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.

[13] M. Ranzato, F. J. Huang, Y. Boreau, and Y. LeCun. Unsupervised learning of invariant feature hierarchies with applications to object recognition. In *Proc. of Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2007.

[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] 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.