
Online Learning of Multiple Perceptual Models for Navigation in Unknown Terrain

Greg Grudic, Jane Mulligan, Michael Otte, and Adam Bates

University of Colorado at Boulder, Boulder CO, USA
Jane.Mulligan@Colorado.edu

Summary. Autonomous robots in unknown and unstructured environments must be able to distinguish safe and unsafe terrain in order to navigate effectively. Stereo depth data is effective in the near field, but agents should also be able to observe and learn perceptual models for identifying traversable surfaces and obstacles in the far field. As the robot passes through the environment however, the appearance of ground plane and obstacles may vary, for example in open fields versus tree cover or paved versus gravel or dirt tracks. In this paper we describe a working robot navigation system based primarily on colour imaging, which learns sets of fast, efficient density-based models online. As the robot moves through the environment the system chooses whether to apply current models, discard inappropriate models or acquire new ones. These models operate on complex natural images and are acquired and used in real time as the robot navigates.

1 Introduction

The system described in this paper operates on an autonomous wheeled vehicle designed for the DARPA Learning Applied to Ground Robots (LAGR) program (Fig. 1). The goal of the program is to apply Machine Learning techniques to autonomous navigation in unknown, unstructured terrains [8].



Fig. 1. DARPA LAGR platform.

The LAGR platform is equipped with Stereo camera systems, but camera resolutions and geometry constrain the resolution and discrimination of

stereo depth data making it useful to about 10m. As a result considerable effort by program participants has been devoted to using image colour and texture for classification of “traversable terrain” and obstacles in the far field. Frequently a standard Learning approach such as Neural Nets, SVM or Decision Trees [5, 7, 1] is applied to build a single model for the terrain at a particular test site. This approach has several drawbacks. First the models are large and difficult to adapt online when terrain appearance changes: a single model cannot capture the diversity of terrain necessary for far field navigation. The models require examples of traversable and non-traversable terrain, where often one may only be able to confidently identify traversable regions. Finally, models are blindly applied across the image, even in regions sufficiently dissimilar to the training set to make their results meaningless. The models offer no means to identify where their classifications can be applied with high confidence.

Extensive work has been done on autonomous driving in structured environments such as highway settings [4]. The work described here instead deals with autonomous navigation in natural outdoor environments where paths and traversable surfaces are not distinctively marked. In contrast with systems which depend on detailed reconstruction of a rover’s environment [2], our system exploits reliable Stereo depth readings in the near field to identify the image properties associated with obstacles and traversable surfaces. These image samples are used to learn single class density models which respond to obstacle or traversable regions in the far field. Our approach builds a set of models as it traverses a new environment. At each frame the current set of models is applied to the image, and if none explains the data a new model is constructed and added to the set. This allows the robot to pass across varied surfaces avoiding obstacles which differ from one area to the next, as when moving from a rocky open field to a path with tree cover. Because we construct separate single class density-based models for traversable and non-traversable terrain, models respond only to terrain that is similar to the training data from which they were constructed.

Online learning of multiple models has not yet been used extensively in Robotics. Bredeche et al. [3] describe an object identification system which learns to label 3 kinds of objects in images based on a human operator’s labeling of an initial image set. Rule-based classifiers are built for HSV image features over a range of scales and structures. As the robot runs the model outcomes are combined based on weights determined by their performance on the validation set. After a fixed number of frames, new models are constructed and the old models evaluated, and possibly replaced, based on their performance over the sequence. Our system uses only Stereo readings to provide learning and validation data for modeling the appearance of obstacles and traversable terrain. New models are constructed whenever existing models fail to agree with near-field Stereo data.

This notion of *concept drift*, where a target (such as the appearance of traversable surfaces) varies over time has been more extensively examined

in the Machine Learning community [9, 6, 11]. Most of this work addresses theoretical issues of bounds on performance based on assumptions about issues such as the rate of drift. They do not however offer a structured approach to questions such as which models to apply to the current sample, when to add new models or discard old ones.

The main questions when maintaining a set of models for a changing environment are: how to determine when the current models are no longer effective, and thus when to construct new models; and for large numbers of models, how to choose which to evaluate at a particular instant.

2 System Overview

The LAGR platform has two “eye” computers each devoted to a separate Stereo rig and associated image processing. A third computer handles planning and control functions. The main issues for our work are how to construct the image costmaps which classify near and far field regions as traversable and non-traversable, and how these are used to plan appropriate behaviour.

2.1 Features

An important question in appearance-based terrain classification is what image features should our model construction be based on? We have experimented with a number of colour encodings and local feature descriptors for classifying traversable and non-traversable terrain in natural outdoor scenes. Currently Normalized RGB ($R/(R+G+B)$, $G/(R+G+B)$, $B/(R+G+B)$) has proven to be the best performer in terms of speed of calculation and classification accuracy. We have also found 1D colour histograms (based on non-normalized RGB) effective for disambiguating terrain classes, but these are much slower to compute. We continue to pursue a separate project to propose and evaluate features for our classification task.

2.2 Combining Cost Maps

The current system generates cost masks based on Stereo and learned appearance-based terrain classification. Initially these masks are the same size as the image, but are mapped to the robot’s local groundplane grid before they are passed to the planner.

For each new frame a feature image is constructed and appearance based classifiers are applied. Two cost masks are constructed, one for obstacles and the other for traversable surfaces. When classifying regions of ground plane using a single class classifier we can only be sure that areas that are classified as in the class are in the ground plane. This doesn’t mean that the non classified regions are not in the ground plane. A second mask that classifies obstacle

regions is needed. This creates a check for the non-classified regions in each mask. If a point in the ground plane mask has a low probability of being in the ground plane then the decision as to whether that pixel is an obstacle is left to the obstacle classifier. This is also true for the obstacle mask. Therefore the assumption that non-classified pixels are obstacles is only made when the ground plane classification is low and the obstacle classification is high.

The two cost masks must be combined in a meaningful way. We take the difference between the obstacle mask O and the groundplane mask G , $O - G$ (scaled from -1 to 1). When the two cost masks disagree, i.e. ground plane mask equals 1 and obstacle mask equals 1 then a 0 is returned. When the cost masks agree, i.e. ground plane mask equals 1 and the obstacle mask equals 0 then a -1 is returned. This means that when sent to the planner the cost in that location will be reduced. When the planner is sent a positive value, the cost in that location is increased.

2.3 Planning System

The planning system is responsible for accumulating data provided by the vision system, mapping it to the world frame, and then computing a route toward some predetermined goal. In our case, the goal is a GPS coordinate and the route is chosen in order to minimize a specialized cost function.

Our planning system is a modified version of a state-based planner provided by Daniel Lee at the University of Pennsylvania. Behaviour in each state S_t is dictated by events that have occurred during the previous state S_{t-1} , as well as information provided by a global state S_{Global} . Information from S_{Global} is usually related to outside sources (bumper hits, human input, messages from the eyes, etc.), while information from S_{t-1} pertains to tasks that the planner has scheduled for itself (i.e. the calculation of a new path or movement of a certain distance). S_{Global} also assumes responsibility for placing cost information into a persistent global cost map MAP_{Global} .

MAP_{Global} is a 2D bird's eye view of the area that the robot has previously sensed and/or can currently perceive. In practice, the map is stored in a 2-dimensional occupancy grid that maintains the relative spatial location of real-world cost information. The occupancy grid has a granularity of 0.5 meters, and contains cost information from the Stereo and appearance-based cost maps. These two forms of information can be weighted differently depending on the navigation task defined at runtime (i.e. learned models will be more useful than stereo data if the task involves water avoidance or navigation through tall grass). The costs in the occupancy grid always fall on a spectrum of belief about the navigability of their associated regions in the real-world. High cost means that an area is less navigable than low cost.

Most of the planner's time is spent finding a path through MAP_{Global} from the current location to the goal. It searches for a path using the A^* search algorithm, where a path is composed of a set of adjacent grid elements of MAP_{Global} . The cost function should reflect some combination of navigability

and distance. A principled way to optimize over these two variables is to represent low navigability as a force that the robot must overcome; and then minimize the amount of work that the robot must do in order to reach the goal. Thus, the cost function to minimize is given by:

$$\text{Navigable Work} = C_1 D_1 + C_2 D_2 + \dots + C_{\text{goal}} D_{\text{goal}}$$

Where C_i and D_i represent the cost and length of path segment i , respectively. In practice C_i values are normalized between 1 and 10 to accommodate A^* . Once a path to the goal has been found, the planner interacts with the robot's low level motors and servos to navigate through the real-world until there is a change in state or until a predetermined amount of time has passed. In the latter case, the planner reenters StatePlan and the process repeats itself.

3 The Learning Framework

3.1 Fast Density Models

The approach taken to far field terrain identification is to build two types of density models - one for identifying traversable terrain and the second for non-traversable terrain. The traversable terrain density models are constructed by sampling image regions that are associated with traversable terrain, and similarly for non-traversable terrain density models.

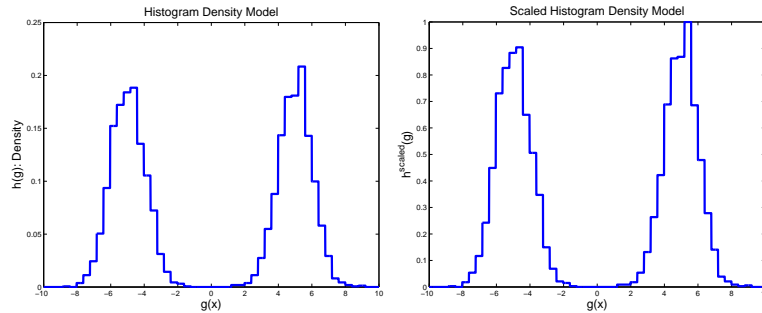
A number of powerful techniques have been proposed in the Machine Learning and Statistics communities for density estimation [10]. Although these techniques can be effective in high dimensional spaces (such as those addressed in this paper), invariably they are too computationally intensive to be applied to real-time robot navigation. In this paper, we present an efficient framework for density estimation that is suitable for real-time terrain classification. Our framework builds many small, fast density models, which are combined to produce a final density model for the entire image.

Assume a set of N examples, each of dimension d , extracted from an image region that is known to be traversable. We symbolize these as $\{\mathbf{x}_1, \dots, \mathbf{x}_N\}$, where each $\mathbf{x}_i \in \mathbb{R}^d$ for all $i \in \{1, \dots, N\}$ is a column vector representing a set of features extracted from an image. Next, the NULL space and the Basis space (i.e. the Principle Components ordered according to relevance) of this data are computed using Singular Value Decomposition. We symbolize the NULL space and Basis space matrices as N_{tr} and B_{tr} respectively (the subscript tr indicates that a traversable model is being constructed). Note that N_{tr} has d rows and m columns, B_{tr} has d rows and k columns, where $k + m = d$ (note that k and d are determined in a standard way based on the floating point precision of the CPU). The NULL space is used to identify when new sensor readings do NOT fall within the scope of the model (i.e. if any new feature vector falls in the null space of the model, the model outputs 0). Therefore,

given some new feature vector \mathbf{x} , the model outputs 0 if $\|N'_{tr}\mathbf{x}\| < \varepsilon$, where ε is determined by the floating point of the CPU.

Next the remaining data is projected into the Basis space B_{tr} , giving $\mathbf{b}_i = B'_{tr}\mathbf{x}_i$, for all $i = 1, \dots, N$, where each $\mathbf{b}_i \in \mathfrak{R}^k$. This data is divided into two approximately equal parts - one set is the training set $\{\mathbf{b}_1^t, \dots, \mathbf{b}_{N_t}^t\}$, and the other is the validation set $\{\mathbf{b}_1^v, \dots, \mathbf{b}_{N_v}^v\}$, where t refers to training data, v refers to the validation data, N_t refers to the number in the training set, and N_v refers to the number in the validation set (note that $N_v + N_t = N$). A linear function is then defined $g_{tr}(\mathbf{x}) = \mathbf{p}'\mathbf{x}$, where \mathbf{p} is the first column of the Basis space B_{tr} , corresponding to the largest eigenvalue of the data PCA space, and thus the coordinate which contains the greatest data range (ie. g_{tr} is the first component of the projected data b_i). Next $g_{tr}(\mathbf{x})$ is converted to a density model using a one dimensional histogram model called $h_{tr}(g_{tr}(\mathbf{x}))$. This histogram density model is constructed by choosing the bin size that maximizes the negative log likelihood of the first dimensions of the validation set $\{\mathbf{b}_1^v, \dots, \mathbf{b}_{N_v}^v\}$. By maximizing the negative log likelihood on the validation data, we ensure that the resulting density model is most representative of the training data. A typical histogram density model is depicted in Figure 2(a). The remaining $k - 1$ dimensions are used to find the minimum and maximum values that $\{\mathbf{b}_1^t, \dots, \mathbf{b}_{N_t}^t\}$, and $\{\mathbf{b}_1^v, \dots, \mathbf{b}_{N_v}^v\}$ have along these coordinates - any new image samples falling outside these ranges result in the model outputting 0. Finally, if training data is available from the non-traversable class, it is used to adjust the density model $h_{tr}(g_{tr}(\mathbf{x}))$ such that the traversable training data is always more probable than the non-traversable data in each bin of the histogram. This ensures that the models are as selective as possible in predicting traversable regions. This algorithm constitutes an efficient and effective framework for bounded density estimation for robotics applications.

The same procedure is used to construct density models for non-traversable terrain, giving $g_{ntr}(\mathbf{x})$ and $h_{ntr}(g_{ntr})$.



(a) Histogram Density Model (b) Scaled Histogram Density Model for Image Segmentation

Fig. 2. Producing Density Models For Image Classification.

3.2 Converting a Density Model to a Segmented Image

Every model $h_{\text{ntr}}(g_{\text{ntr}})$ is used to identify nontraversable terrain in images, and similarly, every model $h_{\text{tr}}(g_{\text{tr}})$ is used to identify traversable terrain in images. This is done by scaling the density models to range between 0 and 1 as in Figure 2(b), which is obtained by taking the density model in Figure 2(a) and dividing by its maximum value. Once this is done, every segment of the image can be classified as shown in Figure 3. Figure 3(a) shows the original image, Figure 3(b) shows the resulting segmentation after passing the feature image through the traversable terrain scaled histogram model $h_{\text{tr}}(g_{\text{tr}})$, and Figure 3(c) shows the resulting segmentation after passing the image through the non-traversable terrain scaled histogram model $h_{\text{ntr}}(g_{\text{ntr}})$. A key property of these scaled density models is that, in every region where the image is dark, the density models have no opinion on whether the region is traversable or non-traversable. This implies that the model is able to predict when image regions are too dissimilar from the data used to construct it, and therefore no predictions can be made using it. As discussed below, this property gives a formal framework for deciding when new models must be added, which models are most appropriate for a given image, and for combining multiple models.

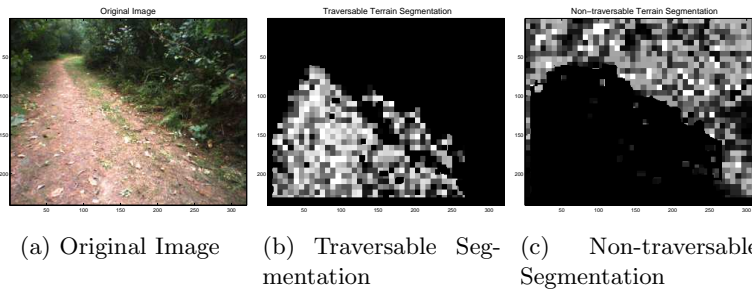


Fig. 3. Using the Scaled Density Models to segment traversable and non-traversable image regions. Image brightness is proportional to confidence in segmentation. Training data was obtained using Stereo in the near range.

3.3 Learning Multiple Models

In the proposed framework, we construct a set of traversable terrain models $\{h_{\text{tr}}^1(g_{\text{tr}}^1), \dots, h_{\text{tr}}^{N_{\text{tr}}}(g_{\text{tr}}^{N_{\text{tr}}})\}$, as well as a set of non-traversable models $\{h_{\text{ntr}}^1(g_{\text{ntr}}^1), \dots, h_{\text{ntr}}^{N_{\text{ntr}}}(g_{\text{ntr}}^{N_{\text{ntr}}})\}$. Each model represents a different type of terrain that the robot encounters as it navigates, and all are constructed in real time. The decision on how many models are needed is made automatically: whenever the current set of models do not explain the observations made by Stereo, new models are added. Thus, if the current set of traversable models do not label a traversable portion of the image as being traversable, a new traversable model is constructed. Similarly, if a region in the image is identified

as non-traversable, and the current non-traversable models do not label this region as such, then a new non-traversable model is constructed to account for this new terrain type. As the robot traverses a variety of different terrains, more models are needed, resulting in potentially thousands of models.

3.4 Choosing Model Subsets

Each density model is representative of a particular terrain type, and as the terrain changes over time, models may not be appropriate for every image. In particular, if a model outputs zero everywhere in an image, then it was constructed from a terrain type that does not appear in the image, and therefore the model should not be used. Similarly if a traversable model has a non-zero response in an image region that Stereo designates NOT traversable, then it should not be used on the current image. An identical argument applies to non-traversable models. Thus the models that are applied to any image are sampled from models that are consistent with near field observation in the image. Even after this selection process, there still may be many hundreds of models that are consistent with an image. In order to ensure real-time operation, not all consistent models can be applied to every image. As a result, the consistent models are ranked according to the magnitude of average response over small random regions of the image. The models with the highest average response are most consistent with the image, while the models with the lowest are least consistent. The first N highest ranked models, where N is dictated by real time considerations, are applied to the image.

3.5 Traversability from Density Models

Once the subset of models is chosen for application to the image, the final traversable image classification is done by taking a maximum over all models, over every region of the image. A similar operation is done for generating the non-traversable classification. The justification for this max operation is that the models that are applied to an image are all consistent with its near field observations. When there is disagreement between the traversable and non-traversable classifications, then the robot errs on the side of exploration and labels the inconsistent parts as traversable, leaving Stereo to correct this possible mislabeling when the robot gets close enough to the region in question. This type of inconsistency is rare because of the way the models are chosen, but can nonetheless occur.

4 Experimental Results

The current set of density models are constructed based on image regions identified as traversable or non-traversable by near-field Stereo. The feature

vectors used are formed from 7 by 7 patches of normalized RGB pixels. Thus, the dimension of each density model constructed is $7 \times 7 \times 3 = 147$.

This system has been applied to a wide variety of real tests (under the DARPA LAGR program) in unstructured outdoor environments. The density models were able to segment difficult terrain in the far field, well beyond stereo range, as demonstrated in Figure 4, where the colour difference between the non-traversable vegetation and the ground is difficult to distinguish. In all the environments tested to date, never more than 47 traversable terrain models, and 43 non-traversable terrain models were required to model the terrain. This demonstrates the flexibility of the individual models. In addition, at each frame up to 20 models were applied (chosen according to the procedure defined Section 3.4) at a rate of about 5 frames per second. The current implementation of this system is written in MATLAB, and we anticipate significant speedups in frame rates when these models are implemented in C. It is also worth noting that systems which classify terrain beyond stereo range have consistently outperformed robots that only use stereo. The system described here also has the benefit of fast online learning and quick adaptation to new environments.

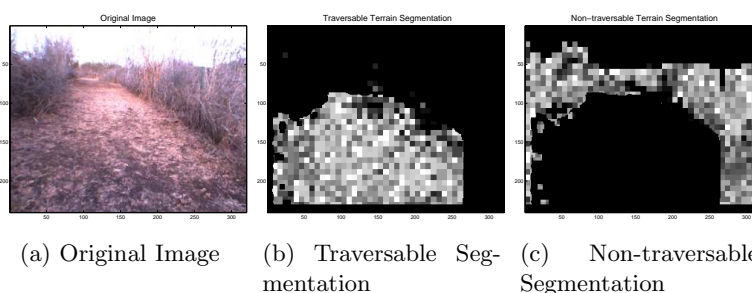


Fig. 4. Using Scaled Density Models to identify traversable and non-traversable image regions. Image brightness is proportional to confidence in classification. Training data was obtained using Stereo in the near range.

5 Conclusion

For autonomous robots traversing unknown terrain, Stereo depth data alone is too “short sighted”. Considerable effort has been devoted to learning models of the appearance of traversable terrain to extend near field Stereo observations of ground plane and obstacles to the far field.

In this paper we describe a successful robot system which learns multiple models of obstacle and ground plane online as the terrain varies over the robot’s trajectory. We describe an efficient way of choosing the best subset of models for each image, out of a set of models, possibly learned over the robot’s lifetime.

The models constructed are density-based and thus respond only to inputs which are sufficiently similar to the data from which they were built. This allows the system to identify novel terrain when its models stop responding, thus triggering the construction of new models. Model construction, selection and evaluation are real-time online operations.

Future efforts include finding optimal image features for terrain classification and allowing each model to identify a preferred feature set for the terrain it encodes.

Acknowledgment

We gratefully acknowledge the equipment and financial support provided by the DARPA LAGR Program (DOD AFRL award no. FA8650-07-C-7702) and NSF CNS-0430593.

References

1. Jim Albus, Roger Bostelman, Tommy Chang, Tsai Hong, Will Shackleford, and Michael Shneier. Learning in a hierarchical control system: 4d/rcs in the darpa lagr program. *Journal of Field Robotics*, 23(11-12), 2006.
2. David Bonnafous, Simon Lacroix, and Thierry Simeon. Motion generation for a rover on rough terrains. In *Proc. International Conference on Intelligent Robotics and Systems*, 2001.
3. Nicolas Bredeche, Jean-Daniel Zucker, and Shi Zhongzhi. Online learning for object identification by a mobile robot. In *Proc. 2003 IEEE International Symposium on Computational Intelligence in Robotics and Automation*, pages 630–635, Kobe, Japan, July 2003.
4. E. D. Dickmanns. Vehicles capable of dynamic vision. In *Proc. 15th Intl. Joint Conf. on Artificial Intelligence (IJCAI-97)*, volume 2, pages 1577–1592, 1997.
5. M. Happold, M. Ollis, and N. Johnson. Enhancing supervised terrain classification with predictive unsupervised learning. In *Proceedings of Robotics: Science and Systems*, Cambridge, USA, June 2006.
6. David Helmbold and Philip Long. Tracking drifting concepts by minimizing disagreements. Technical Report UCSC-CRL-91-26, University of California, Santa Cruz, Santa Cruz, CA, 95064, 1992.
7. Andrew Howard, Michael Turmon, Larry Matthies, Benyang Tang, Anelia Angelova, and Eric Mjolsness. Towards learned traversability for robot navigation: From underfoot to the far field. *Journal of Field Robotics*, 23(11-12), 2006.
8. L. D. Jackel, Eric Krotkov, Michael Perschbacher, Jim Pippine, and Chad Sullivan. The darpa lagr program: Goals, challenges, methodologies, and initial results. *Journal of Field Robotics*, 23(11-12), 2006.
9. Jeremy Z. Kolter and Marcus A. Maloof. Using additive expert ensembles to cope with concept drift. In *Proc. Intl. Conference on Machine Learning*, 2005.
10. Pascal Vincent and Yoshua Bengio. Manifold parzen windows. In S. Thrun S. Becker and K. Obermayer, editors, *Advances in Neural Information Processing Systems 15*, pages 825–832. MIT Press, Cambridge, MA, 2003.
11. Gerhard Widmer and Miroslav Kubat. Learning in the presence of concept drift and hidden contexts. *Machine Learning*, 23:69–101, 1996.