Sensor and Classifier Fusion for Outdoor Obstacle Detection: an Application of Data Fusion To Autonomous Off-Road Navigation

Cristian S. Dima, Nicolas Vandapel and Martial Hebert
Carnegie Mellon University
The Robotics Institute
Pittsburgh, PA 15217, USA
cdima,vandapel,hebert@ri.cmu.edu

Abstract

This paper describes an approach for using several levels of data fusion in the domain of autonomous off-road navigation. We are focusing on outdoor obstacle detection, and we present techniques that leverage on data fusion and machine learning for increasing the reliability of obstacle detection systems.

We are combining color and IR imagery with range information from a laser range finder. We show that in addition to fusing data at the pixel level, performing high level classifier fusion is beneficial in our domain. Our general approach is to use machine learning techniques for automatically deriving effective models of the classes of interest (obstacle and non-obstacle for example). We train classifiers on different subsets of the features we extract from our sensor suite and show how different classifier fusion schemes can be applied for obtaining a multiple classifier system that is more robust than any of the classifiers presented as input.

We present experimental results we obtained on data collected with both the Experimental Unmanned Vehicle (XUV) and a CMU developed robotic vehicle.

1. Introduction

Numerous military and civilian applications call for dependable autonomous vehicles that can navigate off-road. Robotic vehicles can help remove people from dangerous missions, can reduce costs and the time required for deployment. One of the more challenging aspects of autonomous navigation is perception in unstructured or weakly structured outdoor environments such as forests, small dirt roads and terrain covered by tall vegetation. We focus on obstacle detection, where we consider an obstacle to be any region that a vehicle should not attempt to traverse (e.g. humans, trees, big rocks, large holes, large amounts of water). Unfortunately, the difficulty of the problem is such that even human performance in this domain is not perfect.

We believe that in order to achieve acceptable levels of autonomy, vehicles operating in off-road conditions will need to rely on redundancies both at the sensor level and in the decision-making process. Essentially, obstacle detection can be seen as an inference problem: there exists no sensor that will directly indicate if a region in space is an obstacle or not. As a result, we will need to use the available information about such a region to *infer* if it is safe to traverse it or not. Intuitively it should be the case that having more information should lead to better inferences, which translate in turn to higher degrees of reliability of the obstacle detection system.

Another reason for which outdoor navigation should benefit from having several sensing modalities is that their failure modes are often different. Even if a good quality color image can generally provide a lot of information, limitations in the dynamic range of existing cameras make it hard to extract information from images which contain shadows and bright spots, or from images taken at dusk or dawn. A laser range finder is not sensitive to such issues. Similarly, there are times of the day when an infrared camera - which can normally provide great information for detecting humans, water and vegetation - does not produce very useful information. A more diverse set of sensing modalities would increase the chances that at least some of the sensors can produce useful information allowing the autonomous vehicle to pursue its mission.

In addition to data fusion, our approach relies quite heavily on machine learning. Detecting obstacles in environments that are as complex as the ones we are considering requires complex decision schemes which involve large numbers of parameters. Deriving such schemes manually can be an extremely tedious process. We believe that manually "optimizing" the performance of a system with many parameters is not a satisfactory solution.We would like our robots to be easily adaptable to new environments and operating conditions, and for this purpose we will use automated methods for tuning our systems.

Using several sensing modalities or machine learning are certainly not new ideas in the mobile robotics field. A quick look at the previous work shows that sensor fusion has been a constant presence in this area from the earliest mobile robots to the plaforms that define the current state of the art. Begining with the indoor HILARE robot in 1979 [7], Moravec's Stanford Cart and CMU Rover ([14], 1983) and continuing with the outdoors Ground Surveillance Robot [10, 9], the Autonomous Land Vehicle [5], the NAVLAB series of autonomous vehicles [22, 8] and the Demo I-II-III project [23], numerous groups have used sonars, TV cameras, IR sensors, contact switches and laser range finders in order to tackle the obstacle detection problem.

In 1992, Pomerleau [17] demonstrated the first successful application of machine learning methods to the problem of mobile robot navigation. Soon after Davis and Stentz [6] proposed the MAMMOTH system which employed a neural network to learn how to combine steering angles produced by other neural networks using image and laser data.

It is interesting to contrast the machine learning techniques used in early robotic systems such as NAVLAB to more recent approaches such as the Demo III project [1]. While the early systems tried to achieve autonomy by solving one monolithic learning problem (training a neural network to map from grey level images to steering angles in the case of Pomerleau's ALVINN [17]), more recently the trend has been to make intensive use of human domain knowledge and only use learning for those aspects of the problem that are hard to pre-program. For example, in [1] the authors describe a system which uses manually derived rules to identify geometric obstacles, and then filters the results through a color-based classifier that tries to identify the false geometric obstacles caused by vegetation. This latter classifier is trained by fitting a mixture of Gaussians to humanly labeled data.

The approach we propose is located somewhere between the two extremes we just described. We believe that in certain cases it is a good idea to try to go directly from low-level data to an obstacle/non-obstacle decision but we would also like to be able to improve our results by using classifiers produced by human experts. Essentially, we would like to build a "black-box" in which we can feed our data and some other classifiers (trained or pre-programmed, that solve the entire obstacle detection problem or just part of it). The black-box should combine its inputs in such a way that the obstacle/non-obstacle decisions it produces as an output are more likely to be correct than those of any other classifier provided as an input. In this paper we will present results based on several classifier combination techniques and show that such a black-box can be built in practice.

In the following sections we will describe in more detail our problem setup (section 2) and the fusion techniques we have experimented with (section 3). In section 4 we present our experimental results and we draw conclusions and discuss future research directions in section 5.

2. Problem Setup

Considering the large variety of sensing equipment used in outdoor mobile robotics, we will attempt to describe the main assumptions we make about the robotic platform.

Two elements are important for our approach: we assume that the robot is equipped with some form of 3D range sensing (such as a laser range finder or a stereo vision setup), and that it has relatively good pose estimation. The pose estimation requirement can be relaxed, since it is only required for accumulating sensor data over time as the robot moves. In the worst case in which no pose estimation is available, we could still attempt to navigate using a "blindfolded robot" approach: we can ignore all history and make all decisions based on current data.

In addition to range sensing it is frequently the case that robots are equipped with some cameras (e.g. color, black and white, infrared). Our goal will be to combine the range and camera data in order to perform reliable outdoor obstacle detection.

2.1. Data Association

Fusing multisensor data at low-level requires solving the data association problem, which consists of establishing correspondences between the measurements returned by the different sensors. In our case we will need to find such correspondences between our laser data and the images from the color and IR cameras.

The initial step of our calibration procedure consists in determining the intrinsic paramters of our color and IR cameras, for which we use the Matlab Camera Calibration Toolbox ([3]). A relatively simple laser-camera calibration process – consisting of extracting the corners of a checkerboard calibration target in both the laser data and our images – allows us to recover the 3D transformation between the reference frame attached to the laser range finder and the frame of each camera. Using this transformation we can transform all the range measurements from the laser to the camera frames and then use the intrinsic parameters of our cameras in order to find the pixel coordinates where each 3D measurement should project. Thus, for all the laser points that happen to be in the field of view of our cameras we can obtain color and IR information.

Note that if we assume that the position of our robot in a fixed world frame is known we can accumulate laser points expressed in this frame. When a new set of images is captured we can transform these points to the current frame of the cameras and obtain image information for all the accumulated points that are visible.

2.2. Obstacle Detection as a Classification Problem

Assuming that the data association step is completed, there is a choice regarding the space in which we will perform obstacle detection: we can use the 3D space or the image space.

Using a 3D voxel representation for our analysis requires a mapping of the features extracted from images to 3D locations in the world. Using the 3D coordinates of the laser points that project close to a certain location in the image we can map the image properties extracted from that small area to a specific voxel in the 3D representation. The 3D voxels can then be classified as obstacle/non-obstacle voxels using their laser and image-based features.

Performing the analysis in the image space requires the opposite process: one of the images selected as reference is divided into a grid of rectangular patches and all the available 3D measurements are projected into it. Each image patch will contain zero or more laser points, which we can use to extract "laser features" such as range statistics or height in the vehicle frame. The laser features together with the image features (such as texture and color statistics) are the inputs to a classifier which will decide if the image patches as corresponding to an obstacle in the scene or not. The 3D points that project into each patch can be used to estimate the locations of the patches classified as obstacles, a step necessary for obstacle avoidance.

While the two representation models are essentially equivalent, we have chosen to use the image space classification which is more convenient for both labeling data and visualization of the classification results. This is not a limiting factor for the obstacle detection algorithms that we can use in our classifier fusion approach: any labeling of 3D voxels can be converted to a labeling of image patches and vice-versa.

The setup we have presented reduces the problem of obstacle detection to the one of binary classification of image patches in the obstacle/non-obstacle classes. For each image patch we extract color, texture, IR and various laser statistics features which can be used as inputs to our classification methods.

The learning methods used for the experiments presented in this paper are all supervised algorithms. We produce manually labeled data by selecting area of interest in images and classifying them as obstacles or non-obstacles.

3. Classifier Fusion

3.1. Motivation

We have described a method for extracting information (or "features") from several different sensors and using them as inputs to classification algorithms. If we reduced ourselves to simply concatenating all the feature vectors we would essentially perform a simple form of data fusion at the pixel (or more precisely image patch) level. In many mobile robotics applications it is beneficial to be able to also include already existing classifiers that might incorporate significant amounts of domain knowledge. As we have stated in the introduction, we would like to have the capability to automatically learn when to use certain classifiers and how to combine them with and based on the available input data.

The reasons for which classifier combination might be desirable in robotics applications include:

- Several research groups might work on obstacle detection algorithms, making possibly different assumptions about the scene and about the sensors. It is likely that the failure modes of their algorithms will be slightly different, which leads to the question whether by aggregating the decisions of all the classifiers in the pool we could do better on average than any individual algorithm.
- Certain types of obstacles can be particularly difficult to detect: thin wires and negative obstacles (such as holes and trenches) are good examples. While in such cases it might hard to implement a general obstacle detection algorithm that "learns" how to detect them, human understanding of the constraints specific to the obstacle to be detected can lead to much more effective *specialized* detectors. Learning classifier fusion automatically would enable us to determine the weights and rules that should be used with such specialized classifiers without manually tuning parameters based on their false alarms and detection rates.

3.2. Algorithms

In this paper we will discuss three algorithms for classifier combination: committees of experts ([15, 2]), stacked generalization ([27]) and a slight variation of the AdaBoost algorithm ([20]. While our classifier fusion experiments are not limited to these specific algorithms, we consider them to be different enough from each other to be representative for the results one could expect from applying classifier fusion in our domain.

1. **Commitees of Experts**

   Initially described as a method for improving regression estimates in [16, 15], a committee of experts can be used for both regression and classification. The idea behind the algorithm is simple: if we have a pool of L experts that estimate a target function $f(x)$, we can linearly combine their outputs as $f_{COE}(x) = \sum_{i=1}^{L} \alpha_i f_i(x)$, where $f_i(x)$ is the estimate produced by the i^{th} expert. Under this model it can easily be shown [16, 15, 2] that the optimal (in the mean squared error sense) α_i's are given by

   $$\alpha_i = \frac{\sum_{j=1}^{L} (\mathbf{C^{-1}})_{ij}}{\sum_{k=1}^{L} \sum_{j=i}^{L} (\mathbf{C^{-1}})_{kj}}$$

   where $\mathbf{C}$ is the error correlation matrix. It can be shown that the mean squared error of the committee is always smaller than or equal to the average mean squared error over the classifier pool. In fact, if we assume that the experts make uncorrelated zero mean errors the error decreases by at least a factor of L. Obviously, this is overly optimistic: in reality the errors of the classifiers are going to be correlated so the reduction in error will be much smaller. However, given the simplicity of the method it is very attractive to use it. The assumption that needs to be made for the COE fusion approach is that the classifiers in the pool are trying to solve the same problem. As a result, this technique has the limitation of not being able to support specialized classifiers.

2. **Stacked Generalization**

   Introduced by Wolpert in 1990 [27], stacked generalization (or "stacking") was initially presented as a method for combining multiple models learned for classification. Since then, stacking has also been used for regression [4] and even unsupervised learning [24].

   Despite being an extremely simple algorithm, stacked generalization is quite difficult to describe. To make the task easier, we describe what stacked generalization (SG) would be equivalent to if we are willing to assume that a very large amount of training data is available, and then explain the actual algorithm.

   In the form described by Wolpert in [27], stacked generalization is a two stage classifier. Just like in the case of committees of experts we will assume that we have a pool of L trainable experts that estimate a target function $f(x)$. These classifiers are what Wolpert calls the "level-0 generalizers", and are trained in the first stage of SG. The second stage consists of training a classifier that takes as inputs the outputs of the level-0 generalizers and tries to produce the correct label as an output. This classifier is called the "level-1 generalizer", and its purpose is to learn the biases of the level-0 generalizers.

   The crucial element of stacked generalization is that the level-1 generalizer should be trained using data that is "new" to the level-0 generalizers, since we are interested in learning about their generalization properties and not their ability to overfit. In the ideal case where very large amounts of training data were available, this could simply be achieved by splitting the training data and reserving half of it (for example) for training the second stage classifier. The only difference about the stacked generalization algorithm and the method we just described is that in the real algorithm a cross-validation scheme is used so that all the data is used for training both stages of the classifier.

   Stacked generalization works surprisingly well in practice, and it has been applied successfully in other domains such as ATR ([26]).

3. **AdaBoost with Classifier Selection**

   AdaBoost is an algorithm that has been shown to be somewhat similar to the popular support vector machines, in that it tries to maximize the separation margin. Shapire and Freund [20] proposed a clever iterative algorithm that solves the margin maximization problem with the only requirement that a so-called "weak classifier" –a learning algorithm that can perform better than a random one– is available.

   The intuitive idea behind AdaBoost is to train a series of classifiers and to iteratively focus on the hard training examples. The algorithm relies on continuously changing the weights of its training examples so that those that are frequently misclassified get higher and higher weights: this way, new classifiers that are added to the ensemble are more likely to classify those hard examples correctly. Aside from this intuition, AdaBoost's training scheme corresponds to performing gradient descent on an error function that exponentially penalizes small classification margins [13, 21].

   Our small variation to the regular form of Adaboost consists in allowing the algorithm to choose at each iteration which *type* of weak classifier to train. Assuming that we have a pool of classifiers and that some of them can be trained, we allow the algorithm to examine all the classifiers in our pool –training the ones that are trainable– and select the one that can best classify the training examples given their current weight distribution. Thus, AdaBoost will select one of the classifiers available at each iteration.

   Note that while this is not the regular procedure for training AdaBoost, we are not modifying any of the assumptions that the algorithm is based on. A similar application of AdaBoost was successfully demonstrated by Tieu and Viola [25] in the context of automated image retrieval.

4. Experimental Results

4.1. Features

In order to validate the techniques described so far we have performed experiments with both the XUV and another CMU robotic platform. While the two vehicles are equipped with different sensors and have different geometries, we have used the same approach (described in section 2) to extract information about the scenes. For each patch in our image grids we have computed the following features:

- **Color.** The images are converted to the LUV color space; we extract the mean and standard deviation of the pixels in a patch for each channel, obtaining 6 color features.
- **Texture.** The FFT representation of each patch is computed, and it is then divided into 6 bins for frequency and 6 for the orientation. The means and standard deviation of the energy in each bin are computed, resulting in a total of 24 texture features.
- **Infrared.** The mean and standard deviation of the IR pixel values for each patch are computed, resulting in 2 IR features. The correspondence between the color patches (used as reference) and IR patches is established using the 3D information provided by the laser points that project in the color patch.
- **Laser (simple statistics).** Using the laser points that project into each image patch we estimate the average height expressed in the vehicle frame, and the standard deviations in the XYZ directions relative to the vehicle frame. This results in 4 simple laser features.
- **Laser (Vandapel-Hebert features and classification [11]).** As a good example of a specialized classifier we might want to incorporate into our system, we have used an implementation of the technique described in [11] for terrain classification. The method looks at the local point distribution in space and uses a Bayes classifier to produce the probability of belonging to 3 classes - vegetation, solid surface and linear structure. The method takes as input a sparse set of 3-D points. At each point the scatter matrix is computed using a predefined support region. The decomposition in principal components of this matrix leads to the definition of three saliency features characterizing the 3-D points spatial distribution as ”random”, ”linear” and ”surface”. We use both these saliencies and the probabilities of belonging to each class, which results in a number of 6 features. We will refer to these features as “Laser VH”.

4.2. Experiment 1

The first experiment we will present is based on data collected with the XUV robotic platform. The vehicle is equipped with a laser range finder unit, two 640x480 Sony color cameras and an infrared camera with the same resolution. The laser unit and the cameras are mounted inside a pan-tilt platform.

We have evaluated the performance of the various feature sets and the benefit of the different fusion strategies by attempting to solve a problem that is very important for outdoor mobile robotics: detecting dirt roads. While the road detection is not an instance of an obstacle detection problem, notice that our setup is essentially solving binary classification problems and as such can also be used for terrain classification.

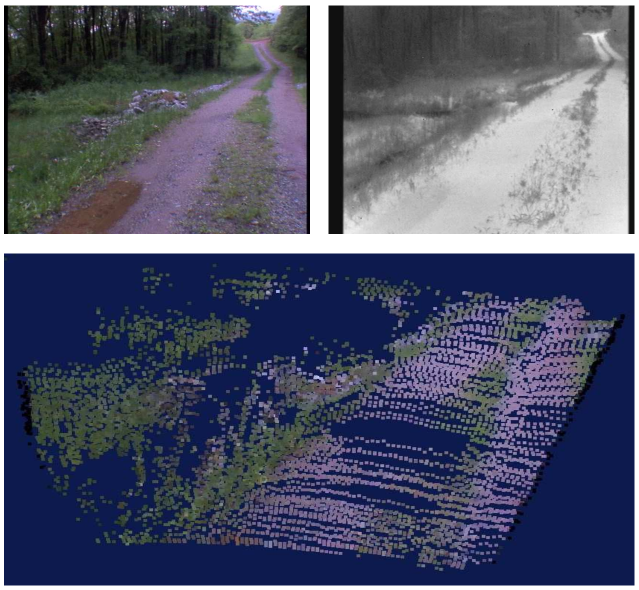

Figure 1. A typical scene from the road detection dataset: the color image (top-left), the IR image (top-right), the 3D point cloud in which points are colorized based on the color image (bottom).

The data logs used for this experiment were collected at the ARL Fort Indiantown Gap robotics facility. Each data log contained color and infrared images, together with vehicle position and range data from the vehicle. We have used 3 independent datasets (2 merged into the training set, 1 used as a test set). The corresponding images were manually labeled in the two classes of interest. We have only used image patches that contained laser points, which resulted in 18963/8582 patches in the train/test datasets. The percentage of road patches was 0.62/0.63.

After labeling the data and extracting the features we have trained several classifiers on this problem. More specifically, we compared the performance of neural networks trained on subsets of our full feature vector (such as color, texture, IR, laser simple and laser VH) with the performance of a neural network that has access to the full vector. We also compared their performance to two of our classifier fusion algorithms, stacked generalization and committees of experts. The numerical results are presented in Figure 2, while Figure 3 presents a graphical representation of the average error rates.

Name	Mean	Std Dev
SG	2.89	0.44
CoE	3.77	0.54
Color	9.45	2.79
Texture	28.73	2.02
IR	12.33	5.22
Laser Simple	17.33	5.29
Laser VH	11.72	3.13
All Features	3.19	0.61

Figure 2. Error rates for the road detection experiments. From the first row down we have stacked generalization, committees of experts, and color, texture, infrared, laser simple, laser VH, and all feature based neural networks.

In order to estimate the error rates and standard deviations we performed 10 fold cross-validation without prior randomization of the patches. We chose not to use randomization in order to avoid getting overly optimistic results: since there is high degree of correlation between neighboring image patches, splitting them randomly would lead to unrealistic similarities between the training and testing datasets. We have also performed experiments with completely separate training and test datasets (i.e. without cross-validation) and the error rates we obtained were similar to the ones produced by cross-validation.

Overall our results are encouraging: they confirm that performing both low-level data fusion and classifier fusion can significantly improve classification performance. The fact that committees of experts and stacked generalization performed as well as a neural network that has access to the full feature vector is very positive. While in this case we had full access to all the features (including the ones pro-

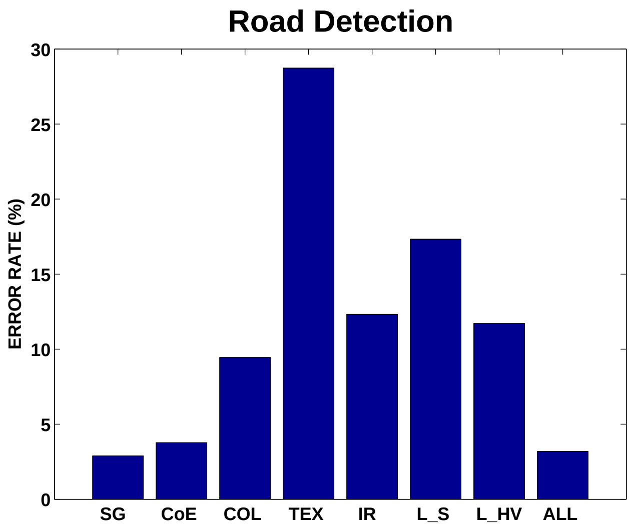

Figure 3. Error rates on the road detection problem. The bars represent in order: (SG) Stacked Generalization, (CoE) committees of experts, (COL) Color, (TEX) Texture, (IR) Infrared, (L_S) Laser simple, (L_VH) Laser VH and (ALL) all features based neural networks.

duced by the VH classifier) which reduces the importance of classifier fusion, it is important to confirm that algorithms like COE and SG can learn to combine input classifiers very effectively.

It is interesting to notice that the VH features (which effectively represent a form of specialized classifier) perform significantly better than the simple laser statistics, despite the fact that exactly the same laser points are used as inputs in both cases. This is a perfect example of why one would like to be able to fuse several classifiers.

4.3. Experiment 2

The second experiment we present uses data collected with a CMU developed robotic platform (a large tractor). The vehicle is equipped with two Sony DFW-SX900 high-resolution color digital cameras producing 1280x960 images and two laser range finder units which are based on mechanically scanned SICK LMS units. At the time the data logs were recorded the vehicle did not have an IR camera.

The experiment we performed on CMU data used the same types of features as the ones based on XUV data, except for the laser VH and the IR features which were not available. The cameras and the laser units have performance characteristics that are quite different from those of the XUV sensors. This makes the experiment even more interesting: we are claiming that using automated learning makes our fusion techniques applicable to many different vehicles and sensor configurations. This is an example of such an application of the same techniques fusion techniques on significantly different vehicles.

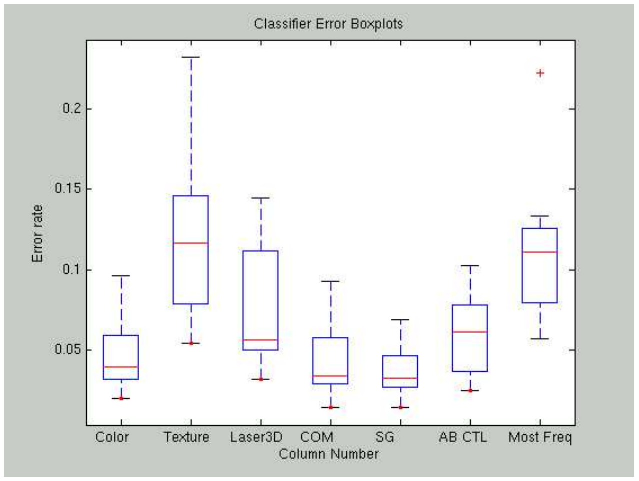

Figure 4. Box plots representing the classification performance on the obstacle detection problem. The rectangle for each classifier represents the interquartile range and the horizontal line is the median. From left to right we have the color, texture and laser based classifiers, the committee of experts (COM), stacked generalization (SG), AdaBoost (AB CTL) and Most Frequent, a classifier that always predicts the most frequent class without using any features.

The problem we attempted to solve in this case was obstacle detection, using a dataset in which the obstacle was a human walking in front of the moving vehicle in an area with tall vegetation. To make the problem non-trivial the human was wearing a camouflage jacket. The raw classifiers were neural networks, this time using color, texture and simple laser features. The classifier fusion strategies we compared were stacked generalization, a committee of experts and the version of AdaBoost we described. The dataset we used contained 22989 non-obstacle and 2893 obstacle image patches (we used 20x20 patches).

The results presented in Figure 4 were obtained performing 10 fold cross-validation on our dataset. Since the two classes (obstacle/non-obstacle) were so unbalanced, we presented the error rate of a "constant" classifier that always predicts the most frequent class. Since only 12 percent of our data represents the obstacle class the reader should be aware that an error rate of 10 percent does not necessarily represent good performance.

In this experiment the color classifier performed extremely well, followed by the laser features and the texture which was mostly irrelevant. The explanation is that the vegetation was slightly dry, which made the color of the camouflaged jacket different from the background. Stacked generalization and the committee of experts were able to learn to focus on the color-based predictions and to use the laser information to slightly improve upon the color performance. A t-test based on our cross-validation data showed this slight improvement to be statistically significant.

The boosting algorithm performed slightly worse than the best input classifier. Our analysis indicated that the problem lies in the exponential penalty that AdaBoost "charges" for small classification margins. The algorithm focuses on increasing the margin on a small number of very difficult training examples while actually reducing the margin of the others; as a result, its generalization performance is reduced. A solution to this problem would be to use "soft-margin" AdaBoost variations such as the one described in [18].

5. Conclusions

We have presented a system that uses multisensor data fusion at both the pixel level and the classifier level in order to improve obstacle detection performance for outdoor mobile robots. Our experiments –on different platforms, sensors and feature configurations– confirm the intuition that combining data from multiple sensing modalities can dramatically improve classification performance. Furthermore, we have shown that automatically combining different classifiers in order to leverage on their particular strengths and provide performance that is better than that of any classifier in the pool is possible. We anticipate that this type of approach will have important applications in mobile robotics. We will continue our experiments in order to analyze the performance of our system on different classification problems and with more complex classifier combination schemes such as hierarchical mixtures of experts [12].

The weakest link of our current setup is the fact that we rely on supervised learning. Labeling data for large scale problems is tedious and expensive, and we are currently developing active learning solutions for alleviating the data labeling requirements. The main direction of our effort is to adapt anomaly detection techniques from the data mining field to our domain, but we are also experimenting with methods such as the one described in [19] to iteratively select the next "most informative" data to label. Since in most robotics applications it is usually inexpensive to collect very large amounts of unlabeled data, we believe that active learning has the potential to open numerous new possibilities for the successful application of machine learning in robotics.

6. Acknowledgements

We would like to acknowledge the valuable support of Carl Wellington in developing some of the infrastructure used for these experiments.

This paper was prepared through collaborative participation in the Robotics Consortium sponsored by the U. S. Army Research Laboratory under the Collaborative Technology Alliance Program, Cooperative Agreement DAAD19-01-2-0012. The U. S. Government is authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation thereon.

References

[1] P. Belluta, R. Manduchi, L. Matthies, K. Owens, and A. Rankin. Terrain perception for DEMO III. In *Proceedings of the IEEE Intelligent Vehicles Symposium*, pages 326–331, October 2000.

[2] C. M. Bishop. *Neural Networks for Pattern Recognition*. Oxford University Press, 1997.

[3] J.-Y. Bouguet. Camera calibration toolbox for matlab. http://www.vision.caltech.edu/bouguetj/calib_doc/.

[4] L. Breiman. Stacked regressions. *Machine Learning*, 24(1):49–64, July 1996.

[5] M. Daily, J. Harris, D. Keirsey, K. Olin, D. Payton, K. Reiser, J. Rosenblatt, D. Tseng, and V. Wong. Autonomous cross-country navigation with the ALV. In *Proceedings of the International Conference on Robotics and Automation*, pages 718–726, 1988.

[6] I. Davis and A. Stentz. Sensor fusion for autonomous outdoor navigation using neural networks. In *Proceedings of the IEEE International Conference On Intelligent Robotic Systems*, volume 3, pages 338–343, August 1995.

[7] A. de Saint Vincent. A 3-D perception system for the mobile robot HILAIRE. In *Proc. IEEE Int. Conf. Robotics and Automat.*, pages 1105–1111, San Francisco, 1986.

[8] Y. Goto and A. Stentz. The CMU system for mobile robot navigation. In *Proc. IEEE Int. Conf. Robotics and Automat.*, pages 99–105, Raleigh, North Carolina, 1987.

[9] S. Harmon, G. Bianchini, and B. Pinz. Sensor data fusion through a distributed blackboard. In *Proc. IEEE Int. Conf. Robotics and Automat.*, pages 1449–1454, San Francisco, 1986.

[10] S. Y. Harmon. The ground surveillance robot (GSR): An autonomous vehicle designed to transit unknown terrain. *IEEE Journal of Robotics and Automation*, RA-3(3):266–279, June 1987.

[11] M. Hebert and N. Vandapel. Terrain classifi cation techniques from ladar data for autonomous navigation. In *Collaborative Technology Alliance Workshop*, 2003.

[12] M. I. Jordan and R. A. Jacobs. Hierarchical mixtures of experts and the em algorithm. *Neural Computation*, (6):181–214, 1994.

[13] L. Mason, J. Baxter, P. Bartlett, and M. Frean. Boosting algorithms as gradient descent. In S. Solla, T. Leen, and K.-R. Muller, editors, *Advances in Neural Information Processing Systems*, volume 12, pages 512–518. MIT Press, 2000.

[14] H. P. Moravec. The Stanford Cart and the CMU Rover. In *Proceedings of the IEEE*, volume 71, pages 872–884, 1983.

[15] M. P. Perrone. *Improving Regression Estimation: Averaging Methods for Variance Reduction with Extensions to General Convex Measure Optimization*. PhD thesis, Brown University, May 1993.

[16] M. P. Perrone and L. N. Cooper. When networks disagree: Ensemble methods for hybrid neural networks. In R. J. Mammone, editor, *Neural Networks for Speech and Image Processing*, pages 126–142. Chapman-Hall, 1993.

[17] D. Pomerleau. Progress in neural network-based vision for autonoumous robot driving. In *Proc. of the Intelligent Vehicles '92 Symposium*, pages 391–396, 1992.

[18] G. Rätsch, T. Onoda, and K.-R. Müller. Soft margins for AdaBoost. *Machine Learning*, 42(3):287–320, Mar. 2001.

[19] N. Roy and A. McCallum. Toward optimal active learning through monte carlo estimation of error reduction. In *Proceedings of the International Conference on Machine Learning*, June 2001.

[20] R. E. Schapire. A brief introduction to boosting. In *Proceedings of the Sixteenth International Joint Conference on Artifi cial Intelligence 1999*, 1999. read.

[21] R. E. Schapire. The boosting approach to machine learning. MSRI Workshop on Nonlinear Estimation and Classifi -cation, 2002.

[22] S. Shafer, A. Stentz, and C. Thorpe. An architecture for sensory fusion in a mobile robot. In *Proc. IEEE Int. Conf. Robotics and Automat.*, pages 2002–2011, San Francisco, 1986.

[23] C. M. Shoemaker and J. A. Bornstein. The Demo III UGV program: A testbed for autonomous navigation research. In *Proceedings of the 1998 IEEE ISIC/CIRA/ISAS Joint Conference*, pages 644–651, Gaithersburg, MD, 1998.

[24] P. Smyth and D. Wolpert. An evaluation of linerly combining density estimators via stacking. Technical Report 98-25, Information and Computer Science Department, University of California, Irvine, July 1998.

[25] K. Tieu and P. Viola. Boosting image retrieval. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2000.

[26] L.-C. Wang, L. Chan, N. M. Nasrabadi, and S. Der. Combination of two learning algorithms for automatic target recognition. In *Proceedings of the International Conference on Image Processing*, volume 1, pages 881–884, October 1997.

[27] D. H. Wolpert. Stacked generalization. Technical Report LA-UR-90-3460, Los Alamos, NM, 1990.