Using Binary Classifiers to Augment Stereo Vision for Enhanced Autonomous Robot Navigation

Michael J. Procopio, Thomas Strohmann, Adam R. Bates, Greg Grudic, and Jane Mulligan

University of Colorado at Boulder
Technical Report CU-CS-1027-07

April 2007

Department of Computer Science
Campus Box 430
University of Colorado
Boulder, Colorado 80309-0430

(This page intentionally left blank.)

Using Binary Classifiers to Augment Stereo Vision for Enhanced Autonomous Robot Navigation

Michael J. Procopio, Thomas Strohmann, Adam R. Bates, Greg Grudic, and Jane Mulligan

***Abstract*— Autonomous robot navigation in unstructured outdoor environments is a challenging area of active research. At the core of this navigation task lies the concept of identifying safe, traversable paths which allow the robot to progress toward a goal. Stereo vision is frequently exploited for autonomous navigation, but has limitations in terms of its density and accuracy in the far field. This paper describes image classification techniques which augment near field stereo to identify safe terrain and obstacles in the far field. Machine Learning classification techniques using appearance-based features appear well suited to the task of far-field obstacle detection, where stereo vision fails. In particular, binary classifiers are appropriate for this task and have performance characteristics suitable for real-time navigation systems. In this paper, we examine the use of stereo vision to identify obstacles and safe terrain in the near field, then using the appearance of these identified regions from the image to classify the remaining far field regions. We rigorously evaluate five binary classifiers as applied to the problem for logged image and navigation data and report on their performance. We also perform live experiments on a DARPA LAGR robot and show that the use of image classification techniques to augment stereo vision results in an enhanced navigational capability in the far field.**

I. INTRODUCTION

When humans hike up a mountainous trail, they are usually able to identify the path from the non-path as well as nearby obstacles with little difficulty. In an instant, a person can visually parse the scene and lay out the exact path that they plan to travel. For example, in Fig. 1 above, few would have trouble identifying the primary path. What mechanisms allow for this feat, and how can it be replicated in an algorithmic sense?

Although public victories such as the successful completion of the 2005 DARPA Grand Challenge [1] would seem to indicate the outdoor navigation problem is solved, in reality there are many more problems to be addressed. One key factor relates to the availability and use of GPS coordinates; the challenge posed by navigating through offroad terrain to a GPS waypoint several hundred meters in the distance is much more difficult than navigating along a route defined by frequent GPS waypoints.

The first author gratefully acknowledges the support of Sandia National Laboratories in sponsoring his dissertation research. Sandia National Laboratories is a multiprogram laboratory operated by Sandia Corporation, a Lockheed Martin Company, for the United States Department of Energy under contract DE-AC04-94AL85000

This work was partially supported by NSF IIS-0535269 and NSF CNS-0430593

Robot hardware and support provided by the DARPA LAGR Program

The authors belong to the Computer Science Dept., University of Colorado at Boulder, Boulder, CO

Fig. 1. Natural trail, easily identified by humans and animals (left). DARPA LAGR platform (right).

Stereo Vision for navigation has a long history [2], [3], [4], particularly for offroad navigation where engineered structure such as lines, signs and pavement are not available to add constraint to the problem. In recent years the power of processors and the availability of inexpensive digital cameras has made real-time stereo readily available and fast enough to be able to identify obstacles sufficiently far in advance to navigate at moderate speeds. However, the resolution of typical cameras and the geometry of typical stereo rigs constrain the resolution and discrimination of stereo depth data making it useful to about 10–15m. This can create a very common navigational failure mode involving cul-de-sacs, where the robot gets stuck because it was unable to plan around an obstacle before it got "caught" in it. This is commonly referred to as "near-sighted" behavior. Fig. 2 illustrates this problem; the ideal path is a smooth arc around the edge of the cul-de-sac. Generally, stereo is an effective tool in the near field, but for smooth long range trajectory planning or fast driving we need an approach to understand far field terrain as well.

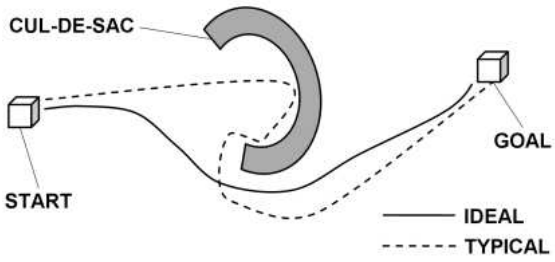


Fig. 2. Effect of stereo *short sightedness* on navigation around a cul-de-sac.

Approaches which use image appearance or color to segment regions of interest for navigation have existed since the 1980s [5], [6], [7], [8]. More recently programs such

as DARPA's Learning Applied to Ground Robots (LAGR) have inspired research into using Machine Learning approaches to exploit image color and texture for classification of "traversable terrain" and obstacles in the far field [9], [10], [11], [12]. A general survey of existing techniques employed in both structured and unstructured outdoor navigation is given in [13], which includes some appearance-based approaches to obstacle avoidance. The techniques addressed therein generally do not incorporate learning and, for terrestrial applications, frequently rely on GPS for both localization and navigation.

This recent interest in classifying traversable and non-traversable regions using only image feature or intensity data leads to the question: which classifiers are effective for this labeling task? More importantly for robotics, can simple linear classifiers which work in real-time effectively identify obstacles and traversable regions? To answer these questions we have undertaken a rigorous analysis of classifier performance for five common classifiers: Linear Kernel Support Vector Machines (SVM), Gaussian Kernel Support Vector Machines [14], [15], K-Nearest Neighbors, Simple Fisher Algorithm and Fisher LDA [16].

The experimental scenario uses near-field stereo data for each frame to provide training examples of traversable ground plane and non-traversable obstacles. These examples are used to train the classifier, and the model obtained is used to label all remaining regions of the image. We evaluate performance both on offline recorded robot logs and in online robot trials. Our results demonstrate that this classification process provides smoother long range trajectories than pure stereo navigation and that supervised learning methods such as linear SVM perform surprisingly well on the labeling task.

II. PROPOSED APPROACH AND MOTIVATION

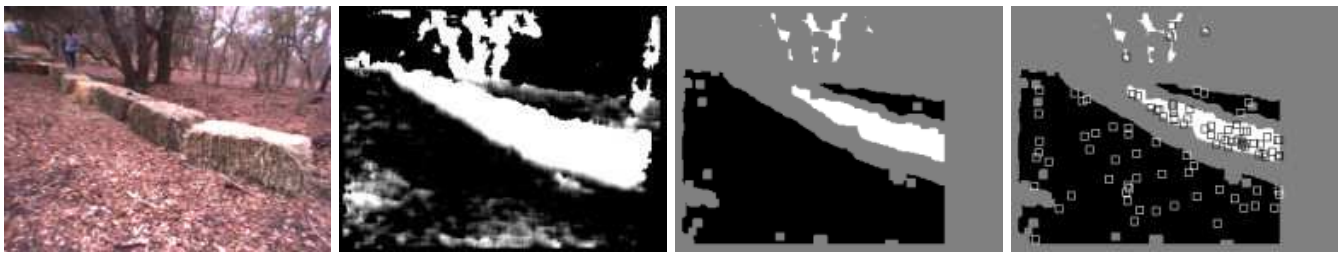
In this paper, we propose that one promising path towards addressing the shortcomings noted above is the augmentation of traditional machine vision approaches with machine learning techniques. Specifically, we use image feature patches taken from images encountered during navigation; these patches are then used as training examples for machine learning algorithms. As initial research in this area, we propose the use of binary classifiers to label traversable and non-traversable terrain inside each image, i.e., one model per frame, based a training set of patches in that same image. This approach addresses each of the main two shortcomings noted in the previous section:

1) By using binary classifiers, a robot is now able to identify obstacles in the far field based on training examples in the near field of the same class. We use stereo classification in the near field to provide the training examples.
2) By creating and storing learned models of traversable and non-traversable terrain, these models can later be recalled and applied to the current scene in order to identify obstacles in the far field without having to first encounter them in the near field. This is long term learning.

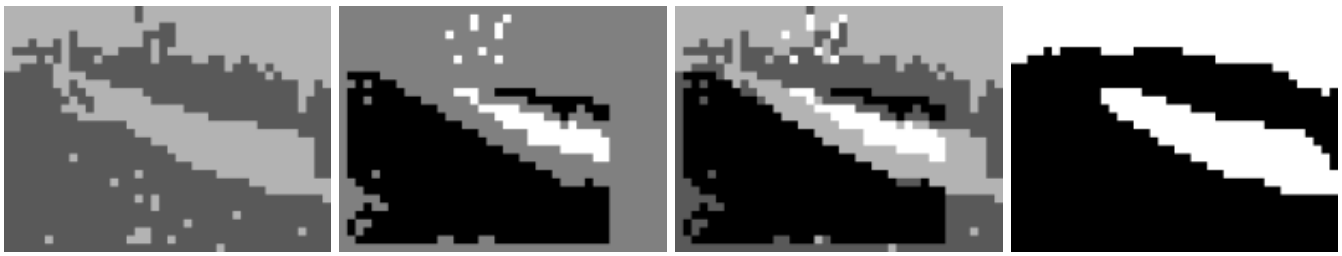
A. Procedure

A detailed description of this approach is outlined below.

1) A pair of RGB images from the robot's stereo cameras is sampled. (Fig. 3(a) shows one such image.)
2) A disparity image is obtained from the stereo pair in step (1) (Fig. 3(b)) [17].
3) From the disparity image, a ground plane / obstacle map (or sample mask) is generated. Thresholds are used to guide this processing, which is based on ground plane deviation. This sample mask is not fully populated with positive and negative examples, because a significant portion of the mask has invalid pixels indicating areas where stereo data was unavailable or the confidence in the stereo reading was not high enough (Fig. 3(c)) [17].
4) From the total set of obstacle pixels appearing in the sample mask, a predetermined number of obstacle training examples (pixels) are randomly selected. The same is done for ground plane examples. If the target number of training examples exceeds the number of available pixels in either class [17], then learning is not performed for this frame (Fig. 3(d)).
5) From the RGB image, a feature image is created. A feature image has the same vertical and horizontal resolution of the main RGB image; however, the actual values for the pixels themselves will be different. Indeed, even the dimensionality of each feature pixel (referred to as the feature depth) can be different; for example, if using texture, a feature image can have perhaps six values per pixel instead of RGB's three (red, green, blue). In these experiments, we use Normalized RGB, so the feature depth is three. If we symbolize (r, g, b) as the raw pixel color values, then the normalized values (r_n, g_n, b_n) are given by:
$$r_n = \frac{r}{r+g+b} \quad g_n = \frac{g}{r+g+b} \quad b_n = \frac{b}{r+g+b}$$
6) Using the pixels selected in the random sets from step (4), a training data matrix is created. The training data consists of vectorized feature data from the feature image extracted from a patch centered around the pixel. This patch is referred to as a window and the window size is a parameter that has implications for both classification and computational performance. Each point in the training data set therefore has dimensionality *d* where *d* is given by:
$$d = \text{window\_height} \times \text{window\_width} \times \text{feature\_depth}$$
A training label matrix is also created by assigning binary output classes to each training point; in this case, by convention we use -1 to indicate ground plane (a negative) and $+1$ to indicate obstacle (a positive).
7) A new model is built for this frame only using the training data from step (6).
8) Test points (windows) in the image are identified. Theoretically it is ideal to perform classification on



(a) RGB Image (b) Disparity Image (c) Labeling Mask (d) Training Examples



(e) Classification Mask (f) Stereo Mask (g) Combined Classification (h) Final Binary Cost

Fig. 3. Obtaining Cost Images

every pixel in the image. In practice, evaluating 50,000 test points through a classifier is too computationally intensive to do at sufficient frame rates; therefore, on the actual robot, a non-overlapping tiling of the image is constructed, and test vectors are extracted from the feature image based on these tiled windows.

9) Each test point is then evaluated in the learned model. The output of this classification (-1 or $+1$) indicates the predicted class (ground plane or obstacle). From the output class of each non-overlapping title identified in step (8), a classification mask is obtained (Fig. 3(e)).
10) At this a point, a tiled cost image needs to be constructed. As a starting point, we take the classification mask from step (9) directly to create the cost image. Optionally, we may wish to copy over the pixels (or tiles) where we have high-confidence stereo information (Fig. 3(f)) and overwrite the classifier output for these areas. If this is performed, the net effect is that learning was performed only for the regions of the image where there was no stereo information (i.e., the invalid or unlabeled pixels in the labeling mask from Fig. 3(c)). In practice, this is usually performed since stereo obstacle identification in the near field is quite robust. Further, we set thresholds in such a way that only very high confidence stereo information is accepted. The combined mask is shown in (Fig. 3(g)).
11) The combined stereo/classification mask from step (10) is then smoothed to create a final binary cost image (Fig. 3(h)).
12) The cost image is then projected into the ground plane [17] creating a cost map for robot navigation, which is in turn sent to the planner [18]. In this manner, terrain classification as output by our method is able to influence the low level navigation of the robot.

III. EXPERIMENTAL DESIGN

A. Classifiers

The selection of the set of binary classifiers to evaluate was driven primarily by performance reasons. Our learning technique is quite computationally intensive in that it involves learning a new model every frame from some 100 to 200 training examples, and then evaluating 520 (if using 12×12 windows) or 1200 (if using 8×8 windows) test points (assuming a 320×240 image). The dimensionality of test and training points when using the normalized RGB feature space is 432 (for 12×12 windows) or 192 (for 8×8 windows). These parameters, coupled with the need to process four or five frames per second, constrains our search to the more computationally lean classifiers. The classifiers evaluated in this paper are listed below.

1) Linear Kernel SVM [14] [15]
2) Gaussian Kernel SVM [14] [15]
3) K-Nearest Neighbors (KNN) [16]
4) The Simple Fisher Algorithm [16]
5) Fisher LDA [16]

B. Datasets

The datasets listed in Table I are used in the evaluation of the classifiers listed above. The datasets span a broad range of course and lighting conditions, containing a number of different types of obstacles (hay bales, trees, dense foliage, people, and equipment), as shown in Fig. 4. All datasets were extracted from log files recorded during LAGR robot testing in the spring of 2006.

C. Procedure

In general, our experiments follow the procedure outlined in Section II. However, for quantitative analysis we need a

TABLE I

DATASETS

Dataset	Description	Frames	Test Points
1	Open Field w/ Hay Bales	220	5,766,386
2	Wooded Canopy w/ Hay Bales	118	3,028,266
3	Light woods w/ Sparse Trees	157	4,033,477
4	Thick Foliage Next to Open Field	94	2,507,191

(a) Dataset 1 (b) Dataset 2

(c) Dataset 3 (d) Dataset 4

Fig. 4. Representative images from datasets used in experiments.

ground truth image against which we can compare the output of the classifier. This ground truth image is the stereo labeling mask from Section II-A, step (3), as shown in Fig. 3(c).

Recall that the labeling mask also identifies invalid pixels (pixels where there was not adequate stereo information to support a high confidence ground plane or obstacle labeling). In these experiments, the invalid pixels are not used as test points since they do not have an associated output class. The set of test points, then, is formed by taking all pixels which are not invalid and which were not identified as training examples (Section II-A, step (4)).

D. Quantitative Analysis

The traditional metric for evaluating machine learning classifiers is classification accuracy. However, in past years it has become clear that this alone is not a robust metric of classifier performance [19]. The use of Receiver Operator Characteristic (or ROC) curves has been embraced by the machine learning community [19]. ROC curves plot the True Positive Rate vs. the False Positive Rate for two binary sets (in the case, the predicted binary set vs. the ground truth binary set); multiple curves from different algorithms are directly comparable in ROC space on the same plot. ROC curves also provide a summary statistic—area under the curve, or AUC—that can be used as a quantitative basis for ranking classifiers. For an overview of ROC curves, and related Precision-Recall (PR) curves, see [20].

TABLE II

OFFLINE RESULTS – CLASSIFICATION ACCURACY (%)

Algorithm	DS1	DS2	DS3	DS4	Combined
100 trn ex./class					
SVM/Gaussian	95.62	96.02	93.68	95.79	95.21
SVM/Linear	95.99	96.09	90.91	94.53	94.44
Simple Fisher	93.70	95.63	93.93	94.09	94.21
Nearest Neighbor	97.65	93.41	79.76	95.26	91.72
Fisher LDA	88.66	89.78	80.02	89.42	86.73
Mean	94.32	94.19	87.66	93.82	—
50 trn ex./class					
SVM/Gaussian	93.79	94.38	93.56	95.42	94.10
SVM/Linear	94.96	95.29	89.19	94.56	93.43
Simple Fisher	93.87	95.36	93.11	94.00	93.98
Nearest Neighbor	96.61	92.35	77.84	94.97	90.53
Fisher LDA	90.34	91.36	83.46	92.41	89.04
Mean	93.91	93.75	87.43	94.27	—

Both classification accuracy (“Accuracy”) and AUC are reported. Further, the ROC curves themselves are provided. Note that for the K-Nearest Neighbors (KNN) algorithm, an ROC curve cannot really be generated since this algorithm does not return a continuous value indicating the signed distance to the decision hyperplane unlike the other classifiers. Therefore, we do not plot an ROC curve for KNN nor do we report AUC for KNN; however, classification accuracy is provided. (Options for using ROC techniques with classifiers that do not return a continuous hyperplane distance are considered in [21].)

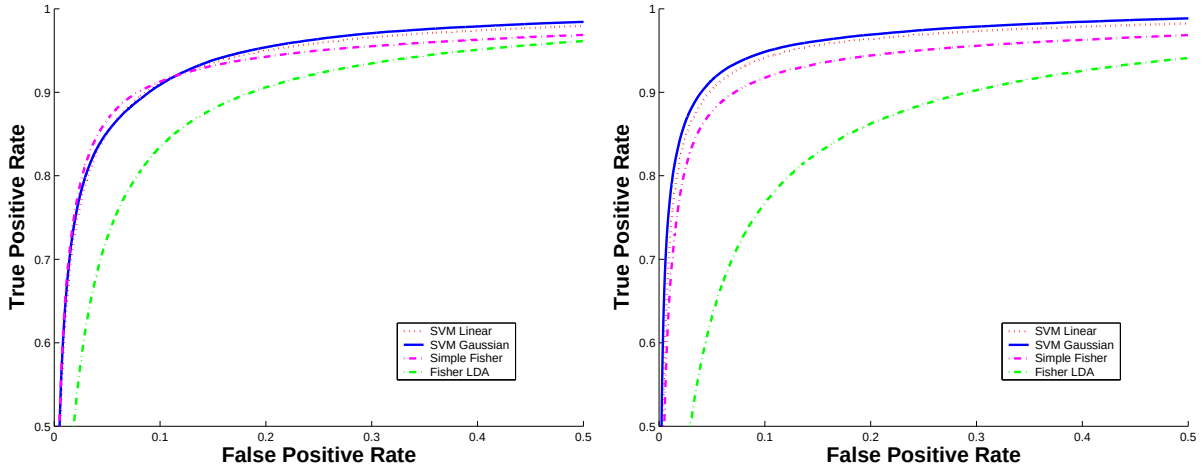
IV. EXPERIMENTAL RESULTS – OFFLINE

Experiments were performed using all classifiers on all datasets. In one experiment, 100 training examples per each class were used. A separate experiment using 50 training examples per class was also performed. The summary of results for classification accuracy is shown in Table II; corresponding AUC results are given in Table III. ROC curves are presented in Fig. 5.

Overall, SVM performed very robustly, with Gaussian Kernel SVM showing the strongest performance. In fact, Gaussian Kernel SVM dominates in ROC space with 100 training examples per class. Interestingly, when using only 50 training examples per class, no one classifier dominates the other in ROC space. The very fast Simple Fisher algorithm performed well in both tests, as did Linear SVM. Their performance vs. computational requirements make them both good candidates for use in a real-time setting.

V. EXPERIMENTAL RESULTS – ROBOT FIELD TEST

Having achieved a ranking of classifiers for this particular problem domain, the expected next step would be to attempt to verify those results by testing our technique on an actual robot in the field and quantifying performance via some metric (time to goal, path distance to goal, etc.). The reality is that the outdoor environments that we test in are very noisy; even subtle lighting changes between runs or minor inconsistencies in robot start location or orientation are



(a) ROC Curve, 50 training examples/class

(b) ROC Curve, 100 training examples/class

Fig. 5. ROC Curves, 50 training examples per class (left) and 100 training examples per class (right).

TABLE III
OFFLINE RESULTS – AREA UNDER ROC CURVE (AUC)

Algorithm	DS1	DS2	DS3	DS4	Combined
100 trn ex./ class					
SVM/Gaussian	96.75	98.62	95.44	99.00	97.49
SVM/Linear	96.65	97.79	95.01	97.50	96.84
Simple Fisher	93.73	97.44	92.82	96.95	95.22
Fisher LDA	89.55	93.37	85.68	90.02	89.55
Mean	94.17	96.80	92.24	95.87	—
50 trn ex./ class					
SVM/Gaussian	94.73	97.60	94.26	98.37	96.14
SVM/Linear	95.42	97.86	92.59	96.93	95.76
Simple Fisher	93.76	97.26	93.01	96.81	95.13
Fisher LDA	92.82	95.28	86.54	95.24	92.44
Mean	94.18	97.01	91.60	96.84	—

a problem. There is enough noise that coming up with repeatable, meaningful test metrics is difficult. The issue is exacerbated in that the subsystems present on the robot (i.e., those used for vision and planning) can be extremely sensitive to even the smallest changes in sensor input. For example, a shadow may cause a false positive classification of an obstacle off in the distance, which is enough to completely change the robot's path even if everything else is constant. This illustrates the problem with using metrics like time to goal or path distance.

We have seen that that binary classification does very well for this problem, with a number of classifiers sharing similarly high performance. Overall, the difference among the top classifiers in offline testing is small enough that any attempt to quantify the advantage of one classifier over the other when running on the actual robot would probably come to be dominated by the noise and variance inherent to the field test setup itself.

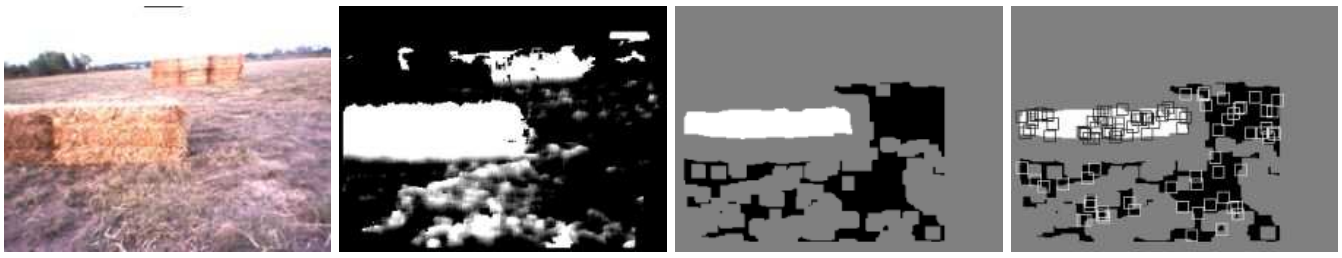
We have therefore adopted a new approach. We want to determine whether or not augmenting near-field stereo vision with far-field learning can have a tangible impact on the robot's navigation. Essentially, we need to demonstrate the effectiveness of learning, and to do so, the following needs to be shown:

1) Using stereo vision alone can result in classic navigational short-sightedness, exemplified by the behavior of going straights towards the goal, drawing near to obstacles along the way and then turning abruptly to avoid them once they come within stereo range (Fig. 2); and
2) Augmenting the stereo vision system with learning in the far field results in smoother, more natural paths only achievable by identifying and planning around distant obstacles.

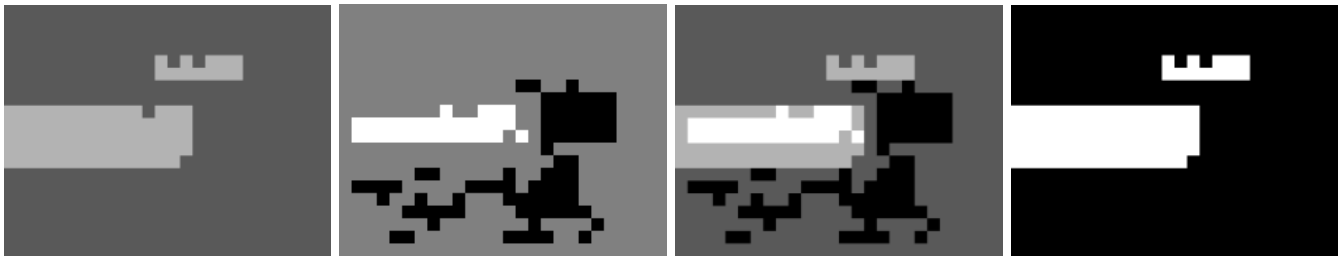
A. Course Design

We constructed a course using hay bales as obstacles that would allow for both (1) and (2) above to be demonstrated. The course consists of two groups of hay bales on the straight-line path from start to goal (Fig. 7). It gives a robot navigating using only stereo techniques ample opportunity to exhibit the classic short-sighted behavior shown in Fig. 2. At the same time, the course allows learning in the far field to impact the robot's navigation, since there are a number of frames in the expected path where the first hay bale group appears in the frame in stereo range, while the second hay bale group appears beyond stereo range off in the distance.

We hypothesize that the learning-augmented system should be able to build models of hay bale obstacles from near-field data and then use these models to classify the second group of hay bales off in the far field. If this procedure is successful, and the far-field obstacles are identified, our expectation is that the robot will plan a smooth path around the distant hay bales and proceed on a smooth trajectory around those obstacles towards the goal. If not, then the robot would continue on a straight line towards the goal after first



(a) RGB Image (b) Disparity Image (c) Labeling Mask (d) Training Examples



(e) Classification Mask (f) Stereo Mask (g) Combined Classification (h) Final Binary Cost

Fig. 6. Frame taken from actual robot field test demonstrating far-field learning from near-field stereo readings.

Fig. 7. Test course (left). Robot on course (right).

avoiding the close hay bale group, putting it on a path to encounter the second hay bale group.

B. Robot Setup

The learning procedure we use on the robot is the same as described earlier in this paper in (Section II-A). We elect to test using Linear Kernel SVM since in our evaluations it offered very high performance with reasonable computational demands. We use a fixed window size of 12×12, which over many tests continues to produce reasonable results; it is a natural filter of false positive obstacle examples but still allows an adequate number of true positives to get through. In order to get an adequate frame rate, we use 50 training examples per class; our offline results indicate a negligible performance decrease from 100 examples per class. This reduced computational demand (which manifests when learning SVM models) allows us to keep our frame rate high enough (approximately 4–5 frames per second) in order to maintain an adequate reaction time.

C. Results

In field testing, we achieved results supporting the hypothesis that augmenting stereo vision with learning in the far field can improve navigational performance. A plot of the robot's paths on the test course from four different runs is shown in Fig. 8. Two runs were stereo-only; two others added learning. This plot was generated from local pose information written to logfiles on the robot. Local pose readings are subject to minor errors due to wheel slippage; these errors can compound over time, but for short distances local pose provides a generally reliable indication of actual path taken.

The stereo runs (dashed lines) are indicative of short-sighted behavior in that (a) they both include abrupt course changes close to obstacles, and (b) they both encountered the second group of obstacles (and show course corrections to avoid them).

The learning-enabled runs performed visibly better. The paths for these runs are far smoother than the stereo paths and are generally closer to the ideal smooth arc from start to goal. They are certainly more natural than the stereo paths and are rather human-like. A collection of images showing the processing of a single frame by the robot on the actual test course is given in Fig. 6. This figure illustrates the identification of the second group of hay bales in the far field as obstacles, as produced by the learned model trained on examples taken from the first group of hay bales in the near field.

Note that while stereo has some notion of the presence of hay bales on the far field (Fig. 6(b)), these readings are not confident enough to be included in the final labeling mask produced from high confidence stereo data (Fig. 6(c)). In contrast to the stereo-only classification shown in Figs. 6(b) and 6(f), the classification mask (output directly from the learned model on a test set of non-overlapped image

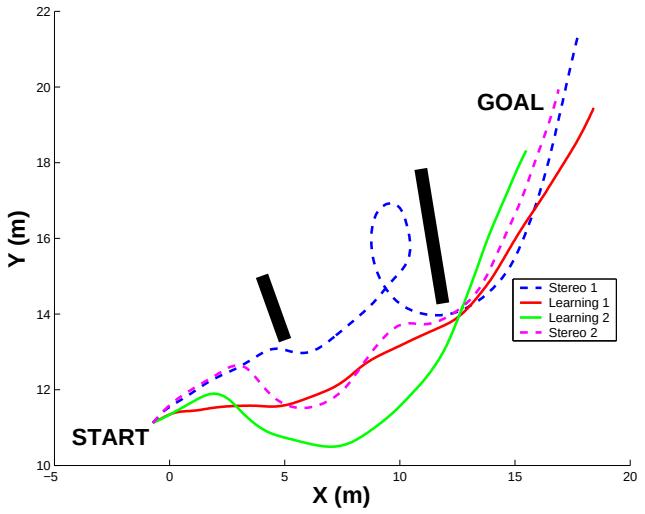


Fig. 8. Robot paths, two stereo only, two with learning.

windows) clearly shows the identification of the second group of hay bales in the far field (Fig. 6(e)).

In the end, this result of smoother driving paths when adding far-field learning to near-field stereo is exciting, if not anticipated, since we know that the planning system of the robot is quite capable of driving the robot along such natural paths (so long as it knows about the far-range obstacles lying ahead). Here, as shown in Fig. 6, the learning system was able to tell the planner about lethal costs in the distance, which were elegantly avoided by planning around them early.

VI. CONCLUSION

This paper addresses the open problem of image-based navigation in unstructured outdoor environments. Confident stereo readings in the near field are used to build models that effectively use image color or other appearance features to classify obstacles and traversable terrain in the far field. The learning and application of these classification models takes place in real time, resulting in a significant improvement in the paths chosen by the robot. We evaluate the performance of five different classifiers: Linear Kernel Support Vector Machines, Gaussian Kernel Support Vector Machines, K-Nearest Neighbors, Simple Fisher Algorithm and Fisher LDA. The best results were obtained with Gaussian SVM, although it was the slowest. Linear SVM and Simple Fisher performed very well and were faster.

Future efforts will address the problem of long term learning, where models constructed by the robot can be stored and used over the robot's lifetime. The main research thrust of this paper has been to demonstrate that such models can be efficiently learned in real time, and can significantly improve robot navigation performance in unstructured outdoor environments.

VII. ACKNOWLEDGMENTS

The authors gratefully acknowledge the contribution of Sandia National Laboratories, the National Science Foundation, and the DARPA LAGR program.

REFERENCES

[1] S. Thrun, M. Montemerlo, H. Dahlkamp, D. Stavens, A. Aron, J. Diebel, P. Fong, J. Gale, M. Halpenny, G. Hoffmann, K. Lau, C. Oakley, M. Palatucci, V. Pratt, P. Stang, S. Strohband, C. Dupont, L.-E. Jendrossek, C. Koelen, C. Markey, C. Rummel, J. van Niekerk, E. Jensen, P. Alessandrini, G. Bradski, B. Davies, S. Ettinger, A. Kaehler, A. Nefian, , and P. Mahoney, "Winning the darpa grand challenge," *Journal of Field Robotics*, 2006.

[2] L. Matthies and P. Grandjean, "Stochastic performance modeling and evaluation of obstacle detectability with imaging range sensors," *IEEE Transactions on Robotics and Automation*, vol. 10, no. 6, Dec. 1994.

[3] D. B. Gennery, "Traversability analysis and path planning for a planetary rover," *Auton. Robots*, vol. 6, no. 2, pp. 131–146, 1999.

[4] D. Murray and J. Little, "Using real-time stereo vision for mobile robot navigation," *Autonomous Robots*, 2000, to Appear (http://www.cs.ubc.ca/spider/little/links/robuds.html).

[5] R. Wallace, K. Matsuzaki, J. Crisman, Y. Goto, J. Webb, and T. Kanade, "Progress in robot road-following," in *Proc. of 1986 IEEE International Conference on Robotics and Automation*, vol. 3, 1986, pp. 1615–1621.

[6] J. Crisman and C. Thorpe, "Unscarf, a color vision system for the detection of unstructured roads," in *Proc. 1991 Int. Conf. on Robotics and Automation*, Sacramento, CA, April 1991, pp. 2496–2501.

[7] D. Kuan, G. Phipps, and A.-C. Hsueh, "Autonomous robotic vehicle road following," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 10, no. 5, pp. 648 – 658, 1988.

[8] M. Turk, D. Morgenthaler, K. Gremban, and M. Marra, "Vits-a vision system for autonomous land vehicle navigation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 10, no. 3, pp. 342–361, May 1988.

[9] 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.

[10] G. Grudic and J. Mulligan, "Outdoor path labeling using polynomial mahalanobis distance," in *Proceedings of Robotics: Science and Systems*, Cambridge, USA, June 2006.

[11] H. Dahlkamp, A. Kaehler, D. Stavens, S. Thrun, and G. Bradski, "Self-supervised monocular road detection in desert terrain," in *Proceedings of Robotics: Science and Systems*, Cambridge, USA, June 2006.

[12] R. Manduchi, A. Castano, A. Talukder, and L. Matthies, "Obstacle detection and terrain classification for autonomous off-road navigation," *Auton. Robots*, vol. 18, no. 1, pp. 81–102, 2005.

[13] G. N. DeSouza and A. C. Kak, "Vision for mobile robot navigation: A survey," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 24, no. 2, pp. 237–267, 2002.

[14] V. Vapnik, *The Nature of Statistical Learning Theory*. New York: Springer, 1995.

[15] C. C. Chang and C. J. Lin, *LIBSVM: a library for support vector machines*, 2001, software available at http://www.csie.ntu.edu.tw/ cjlin/libsvm.

[16] T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning*. New York: Springer, 2001.

[17] D. A. Forsyth and J. Ponce, *Computer Vision A Modern Approach*. Upper Saddle River, NJ: Prentice Hall, 2003.

[18] J.-C. Latombe, *Robot Motion Planning*. Kluwer, 1991.

[19] F. J. Provost, T. Fawcett, and R. Kohavi, "The case against accuracy estimation for comparing induction algorithms," in *ICML '98: Proceedings of the Fifteenth International Conference on Machine Learning*, 1998, pp. 445–453.

[20] J. Davis and M. Goadrich, "The relationship between precision-recall and roc curves," in *ICML '06: Proceedings of the 23rd international conference on Machine learning*, 2006, pp. 233–240.

[21] T. Fawcett, "Roc graphs: Notes and practical considerations for researchers," 2004.