

# Boundary Detection Based on Supervised Learning

Kiho Kwak, Daniel F. Huber, Jeongsook Chae, and Takeo Kanade

**Abstract**—Detecting the boundaries of objects is a key step in separating foreground objects from the background, which is useful for robotics and computer vision applications, such as object detection, recognition, and tracking. We propose a new method for detecting object boundaries using planar laser scanners (LIDARs) and, optionally, co-registered imagery. We formulate boundary detection as a classification problem, in which we estimate whether a boundary exists in the gap between two consecutive range measurements. Features derived from the LIDAR and imagery are used to train a support vector machine (SVM) classifier to label pairs of range measurements as boundary or non-boundary. We compare this approach to an existing boundary detection algorithm that uses dynamically adjusted thresholds. Experiments show that the new method performs better even when only LIDAR features are used, and additional improvement occurs when image-based features are included, too. The new algorithm performs better on difficult boundary cases, such as obliquely viewed objects.

## I. INTRODUCTION

Boundary detection is a central problem in many robotics and computer vision applications. It is a key step in segmenting an object from its background, which is known as figure-ground segmentation. The ability to distinguish a foreground object from the background in a scene can simplify object detection and recognition, and is first step in detecting, modeling, and tracking moving objects in a scene.

An object boundary occurs where one object occludes another from the viewpoint of a sensor. Boundary detection has been studied using static images [1], [2], [3], video [4], [5], and LIDARs [6], [7], [8], [9]. Detecting boundaries is difficult in static images because appearance may change more dramatically within an object than at its boundary. Video sequences can simplify the problem somewhat, since motion artifacts can often be detected at depth discontinuities as the sensor moves.

Three dimensional sensors, such as stereo vision or LIDARs, offer a good alternative to monocular vision approaches for boundary detection. These sensors directly estimate the distance to surfaces in the scene and therefore have the potential to detect depth discontinuities that often appear at object boundaries. LIDARs have the advantage, with respect to stereo vision, of lower uncertainty and reliable performance in untextured regions, generally at the expense of lower angular resolution. However, even the

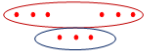



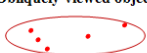

This work was supported by the Agency for Defense Development, Republic of Korea.

Kiho Kwak, Daniel F. Huber, and Takeo Kanade are with Carnegie Mellon University, Pittsburgh, PA 15213, USA {kkwak@cs.cmu.edu, dhuber@ri.cmu.edu, tk@cs.cmu.edu}

Jeongsook Chae is with the Agency for Defense Development, Republic of Korea aelibots@gmail.com

TABLE I

ANALYSIS OF THE POTENTIAL BENEFITS OF LIDAR INFORMATION AND IMAGERY FOR EACH CHALLENGING BOUNDARY DETECTION CONDITION.

| Cases  | Can LIDAR help?   | Can imagery help?                      |
|--|---|--|
| <b>Occlusion</b><br>                       | Yes.<br>Reason about occlusion, orientation of occluded surfaces. | Yes.<br>If appearance is consistent.   |
| <b>Transparent surface / no return</b><br> | Maybe.<br>Use hole filling.                                       | Maybe.<br>If appearance is consistent. |
| <b>Porous objects (e.g., foliage)</b><br>  | Maybe.<br>Analyze variation in range.                             | Maybe.<br>Classify foliage in image.   |
| <b>Long distance objects</b><br>           | Maybe.<br>Can threshold based on distance.                        | Yes.<br>If appearance is consistent.   |
| <b>Obliquely viewed objects</b><br>        | Yes.<br>Adjust threshold based on surface normal.                 | Yes.<br>If appearance is consistent.   |
| <b>Objects close together</b><br>          | No.   | Maybe.<br>But could be unreliable.     |

high-precision range measurements of laser scanners do not trivially solve the boundary detection problem.

A common approach for detecting boundaries using range data is to compute differences in the range of adjacent measurements and to declare any range difference above a given threshold to be an object boundary (see [7] for an overview). While this simple thresholding approach can work well for straightforward cases, it does not perform as well in more challenging scenarios that occur frequently enough to need addressing. For example, if two objects are close together, they may be incorrectly classified as a single object with no boundary between them. We have identified a number of situations that are particularly challenging for boundary detection: occlusions (which can break a single object into multiple objects and create false boundaries) obliquely viewed objects, transparent or translucent surfaces, low-reflectance surfaces, objects that are close together, porous objects (such as trees) and objects observed from long distances. For each of these cases, we considered whether the information from LIDAR could be helpful in detecting such boundaries and also whether imagery could be helpful (Table I). We observe that, while LIDAR can be helpful in many cases, there are some situations where imagery provides complimentary information.

In this paper, we investigate two questions regarding boundary detection. First, can supervised machine learning techniques improve upon the performance of existing threshold-based algorithms for boundary detection using

LIDAR data? Second, can information derived from images provide any additional benefit to LIDAR-only boundary detection techniques? We focus on the problem of boundary detection using a horizontally oriented, single line LIDAR (e.g., SICK LMS291 or Hokuyo UTM-30LX). Single line LIDARs are often used in mobile robot applications for obstacle detection and moving object tracking, and are also used in other applications, including factory safety systems, surveillance, and mobile mapping. When a single line LIDAR is mounted horizontally at an appropriate height, the laser sweeps a horizontal arc, making measurements at constant horizontal angular increments in a plane at an approximately constant height above the ground.

We cast the problem of boundary detection as a classification task. We consider the space between consecutive LIDAR measurements, and predict whether an object boundary exists within this region or not. Given a set of labeled local features derived from the data, standard machine learning tools, such as support vector machines (SVMs), are used to learn the relationship between features computed from the data and the boundary/non-boundary classification labels. The method is fast enough to operate in real-time and straightforward to implement. Furthermore, this framework is general enough that features derived from imagery can be integrated directly as well. The use of images assumes that the camera and the LIDAR are approximately co-located to minimize parallax between the sensors and requires that they are calibrated with respect to one another.

We conducted experiments to evaluate this approach and compared it with a state-of-the-art, threshold-based method. Experiments show that the proposed method improves on this existing boundary-labeling method, especially on the more difficult cases, where the existing method performs poorly. The addition of image-based features further improves boundary detection performance.

The rest of this paper is organized as follows. The next section describes previous research that is relevant to our approach. Section 3 details our boundary classification algorithm. Sections 4 and 5 outline the experimental setup and experimental results respectively. Section 6 concludes the paper and discusses our plans for future work.

## II. RELATED WORK

There is relatively little research on explicitly detecting object boundaries. However, the boundary detection problem can be viewed as the dual of the more commonly studied problem of segmentation, where the goal is to find all the points (or pixels) belonging to a single object. Boundary detection (or segmentation) algorithms can be broadly classified based on whether they use range data or imagery.

The most common method for segmenting range data is to use jump distance thresholds [6], [7], [8], [9]. Boundaries are identified wherever the Euclidean distance between two adjacent points is larger than a threshold. Arras [8] and Spinello [9] observed that although fixed threshold methods work well in indoor environments, they do not work as well in outdoor environments due to the longer ranges involved

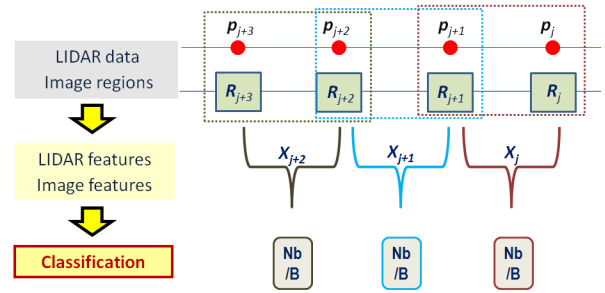


Fig. 1. The boundary detection problem in the framework of classification (Nb: Non-boundary, B: boundary).

and more challenging conditions (e.g., scenarios described in Table I). For example, if the threshold is chosen to be large enough to cluster objects at long distances, then nearby objects are not separated. To solve these problems, new segmentation methods using more complex and adaptive thresholds (typically a function of range) were developed [6], [7]. While these methods address the problem of long distance objects, they still have difficulty with challenging scenarios in outdoor environments.

There has been recent interest in explicit boundary detection using imagery and video. Recently, the Pb boundary estimator [1] and follow-on research [3] have shown promising results on still images. The initial method uses a combination of image gradients and texture-based features in a learning framework to estimate the probability of a boundary existing at each pixel. In follow-on work, improved boundary detection was achieved by combining the original Pb estimator with a global estimate of boundary based on normalized cuts.

Video can also be used to detect object boundaries. Stein *et al.* [4], [5] use motion cues to estimate boundaries in an over-segmentation of the images in a video sequence. The combination of appearance and motion cues results in better detection than appearance alone.

Algorithms that are similar in spirit to ours that learn features from a combination of imagery and LIDAR have been proposed as well [8], [9]. However, these methods are designed for detection and classification of specific objects, such as pedestrians, and, as a result, are not well-suited for the general problem of boundary detection.

## III. BOUNDARY DETECTION AS CLASSIFICATION

We pose the boundary detection problem in the framework of classification (Figure 1). A line-scanning LIDAR produces a sequential series of range measurements with fixed angular increments in polar coordinates. The boundary of an object may lie at any point in this continuum, and, therefore, the true location of the boundary is likely to lie at some position between the measured rays. The classification task is to determine whether an object boundary exists between two consecutive range measurements.

If the scene is observed simultaneously with a camera, then it is possible to calibrate the camera pose with respect to

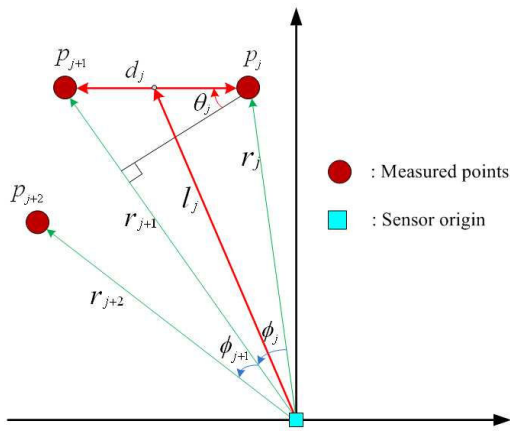


Fig. 2. LIDAR features description.

the LIDAR, which allows the range measurement points to be projected onto the camera image. The centers of projection of the camera and LIDAR need to be as close together as possible to minimize the effects of parallax. Camera image resolution is typically significantly higher than the angular resolution of LIDARs, which means that the region between two projected laser points will contain multiple pixels. Consequently, information from image patches must be aggregated. We use rectangular image patches surrounding the projected LIDAR points for computing image-based features. Intuitively, if the image patches surrounding two points belong to the same object, then the statistics of the image patches are more likely to be similar than if they belong to two different objects.

Evidence of a boundary existing between two LIDAR points may be based on features derived from the range, including distance between real returned points (except intervening points with no return from the laser scanner, which we refer to as "no-return points"), range from origin to mid-point, and surface orientation. Evidence of a boundary between two points in an image is based on image texture features, histogram similarity, mean intensity, standard deviation of intensity. The next two sub-sections describe, the LIDAR-based and image-based features used in our experiments in detail.

Given a set of features derived from the 3D and image data, along with the ground truth labeling of whether a boundary exists between each pair of LIDAR measurements, we use standard machine learning techniques to train a classifier to correctly label novel data queries. We experimented with several types of classifiers, including logistic regression [10] and support vector machines (SVMs) with linear and non-linear kernels [11].

#### A. LIDAR features

When a LIDAR returns consecutive range scan points  $(p_i, i = 1, \dots, n)$  in a full scan, each scanned point can be defined as  $(r_i, \phi_i)$  in the polar coordinates and as  $(x_i = r_i \cos \phi_i, y_i \sin \phi_i)$  in the Cartesian coordinates. The  $n$  denotes the number of scan points, including valid

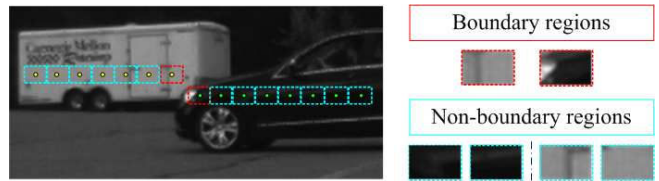


Fig. 3. Left: Examples of image patches used for computing image-based features. Right: Close-ups of some representative pairs of adjacent regions from the left image.

and invalid scanned points in a full scan. A point is considered invalid if the sensor did not receive a return for that measurement, which occurs, for example, when there is no surface within the sensor's range. In this paper, we only consider the relation between valid range measurements  $(p_j, j = 1, \dots, n_r)$ , where  $n_r$  is defined as the number of valid returned points. The LIDAR features are defined as a function  $(f_j : P_j \rightarrow \mathbb{R}^m)$  that takes a pair of consecutive valid points  $(P_j = (p_j, p_{j+1}), j = 1, \dots, (n_r - 1))$  and returns an  $m$ -dimensional value  $(X_j^L = (d_j, l_j, \theta_j)^T, X_j^L \in \mathbb{R}^3)$ . We define the following LIDAR-based features (refer to Figure 2):

1) *Distance between valid returned points ( $d_j$ )*: This feature is the Euclidean distance between points  $p_j$  and  $p_{j+1}$  ( $j = 1, \dots, (n_r - 1)$ ) and is given by

$$d_j = \|p_j - p_{j+1}\|_2 \quad (1)$$

2) *Range from origin to mid-point ( $l_j$ )*: This feature measures the range from the origin to the mid-point between points  $p_j$  and  $p_{j+1}$ , which is given by

$$l_j = \left\| \frac{p_j + p_{j+1}}{2} \right\|_2 \quad (2)$$

3) *Angle between valid returned points ( $\theta_j$ )*: This feature is designed to measure the orientation of the surface with respect to the viewing direction and is given by

$$\theta_j = \arctan \left( \frac{r_{j+1}}{r_j \sin \phi_j} - \cot \phi_j \right) \quad (3)$$

#### B. Image features

Each image feature is based on two rectangular regions,  $R_j$  and  $R_{j+1}$ , extracted from image locations centered on the projection of the LIDAR measurements,  $p_k$  and  $p_{k+1}$  into the image (Figure 3). The image features are defined as a function  $(f_j : P_j \rightarrow \mathbb{R}^m)$  that takes a pair of consecutive valid patches  $(I_j = (R_j, R_{j+1}), j = 1, \dots, (n_r - 1))$  and returns a 3-dimensional value  $(X_j^I = (h_j, m_j, s_j)^T, X_j^I \in \mathbb{R}^3)$ . In our experiments, the size ( $N$ ) of each patch is 11 by 15 pixels.

1) *Histogram intersection ( $h_j$ )* [12]: This feature measures the similarity between two histograms computed over adjacent regions. This is calculated as

$$h_j = \frac{\sum_B \min(H_j, H_{j+1})}{\sum_B H_j} \quad (4)$$

where  $H_j$  and  $H_{j+1}$  are the histograms acquired from the image patches  $R_j$ ,  $R_{j+1}$ , and  $B$  is the number of bins in the histogram. In our experiments,  $B$  is set to 16.

2) *Difference of mean intensity of two patches ( $m_j$ ):* This feature measures the difference of average intensity of adjacent patches

$$m_j = \frac{1}{N} \left( \sum_{m,n} (R_j(m,n) - R_{j+1}(m,n)) \right) \quad (5)$$

3) *Difference of standard deviation of two patches ( $s_j$ ):* This feature returns the difference of standard deviation of the intensity of adjacent patches.

$$s_j = \sigma_j - \sigma_{j+1} \quad (6)$$

where

$$\sigma_j = \sqrt{\frac{1}{N-1} \sum_{m,n} (R_j(m,n) - \mu_j)^2} \quad (7)$$

$\mu_j$  is the mean intensity value of patch  $R_j$ . The value of  $\sigma_{j+1}$  is defined analogously.

#### IV. EXPERIMENTAL SETUP

For our experiments, a SICK LMS-221 LIDAR and a PointGrey Flea2 camera were used to acquire the data sets. The LIDAR has a 180 degree horizontal field of view, with a line scanning frequency of 75 Hz and a 0.5 degree angular resolution. The camera has a 60 degree horizontal field of view, with a frame rate of 30 Hz and a resolution of 800 by 600 pixels. These sensors were mounted on front of a vehicle. The LIDAR was placed at a height of 70 cm, and the camera was installed 30 cm above the LIDAR. The data from each sensor was time-stamped to enable synchronization.

We developed a procedure for calibrating a line scanner with respect to image data. The intrinsic parameters of the camera (e.g., focal length, optical center, pixel aspect ratio, and skew) were computed using the matlab calibration toolbox [13]. The procedure for estimating the extrinsic parameters requires a small set of thin objects to be imaged by the laser scanner and the camera. Furthermore, a device for detecting the laser spot on the thin objects is needed, and for our experiments, we use tripods for the thin objects, and an infrared viewer to locate the laser spots on the tripod legs. By using this method, a calibration of both sensors has been achieved, which enables projecting LIDAR range measurements onto the camera image.

The data for our experiments consists of sequences that include walking and running people and moving vehicles in different directions with respect to the stationary sensor platform, as well as a number of non-moving vehicles and other stationary objects. Data sets were collected during daylight and in different environmental conditions – sunny days and snowy days.

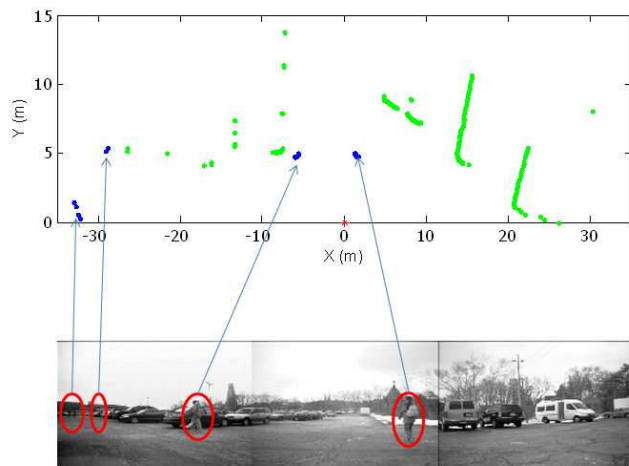


Fig. 4. An example of automatic classification of foreground and background points (top), along with the corresponding images for reference (bottom).

One of the important aspects of training a classifier is to ensure that the input training data is not unduly biased. In our case, a substantial percentage of the measurements within each scan are identical from scan to scan. These boundaries (and also non-boundary points) are part of the non-moving background. If we were to include such points from every scan in the training data, then the algorithm would be biased toward these redundant measurements. Instead, we identify the background regions in the training data and only use these measurements once. The foreground data changes sufficiently between scans, due to the changing position relative to the background and the changing orientation of the objects (e.g., car turning).

We implemented a simple background detection algorithm to automatically identify the background in training data based on a small sample of scans of each scene. The algorithm assumes that the sensing platform is not moving, but it is allowable to have some moving objects within the scene. The algorithm measures the range in each scanning direction (which is constant between scans), and accumulates these ranges in a histogram for each scanning direction. The most frequently occurring ranges above a threshold in frequency are considered to be background points. The remaining points are filtered to remove transient random noise, which can occur due to various sensor artifacts and environmental conditions (like snow), and any unfiltered points are considered to belong to moving foreground objects (Figure 4). The spaces between adjacent background points are then manually labeled as boundary or non-boundary using an interactive labeling tool.

Once the background is identified, new scans can be labeled and divided into foreground and background automatically. Points are labeled as background if they fall within a short distance from the background surface range for the given scanning direction. Any remaining points are filtered as described above, and the boundary between foreground points and background points is implicit based

## V. EXPERIMENTAL RESULTS

Using the framework described above, we performed a series of experiments to evaluate the effectiveness of our boundary detection approach. The experiments include: a) a comparison of different types of classifiers; b) a comparison of our algorithm with a state-of-the-art method that uses dynamic jump distance thresholding; and c) a comparison of our algorithm using LIDAR features, image features, and a combination of LIDAR and image features.

### A. Choice of classifier

We compared the performance of three different classifiers: SVMs using a linear kernel, SVMs using an RBF kernel, and logistic-regression. For the SVM classifier implementation, we used LibSVM [16] and determined the penalty parameter ( $C = 10$ ) for the linear kernel SVM and the penalty and kernel parameters ( $C = 10$ ,  $\gamma = 0.1$ ) for the RBF kernel SVM by cross validation. The results of the experiment show that there is not much differentiation between the performance of the chosen classifiers (Figure 5). In the ensuing experiments, we use the SVM classifier with a linear kernel, which is sufficiently fast and does not suffer from the possibility of overfitting.

### B. Comparison with dynamic thresholding

In our second experiment, we compared the performance of our algorithm – using LIDAR features only – with the adaptive breakpoint detector (ABD) algorithm described by Borges and Aldon [6]. The ABD algorithm uses a threshold that scales as a function of distance from the sensor to determine the boundaries between objects.

We evaluated the two algorithms on the same data sets, which included all instances inside and also outside the camera FOV. The results, shown in Figure 6, show that our algorithm outperforms the ABD algorithm for all threshold values. We also manually extracted a subset of the full data that consisted entirely of challenging cases as described in Table I. The performance of both algorithms on this challenging data set is lower than on the full data set, but our algorithm still outperforms the ABD method. Figure 7 shows an example of the boundary classification results of the two algorithms for a portion of an example scan. In this example, the ABD algorithm breaks up single objects when they are viewed obliquely (for example, a car seen from a slanted angle), whereas our method classifies these instances correctly.

### C. Effect of sensor modality

The next experiment was designed to test the contribution of the LIDAR-based features versus the imagery-based features. In this experiment, only instances within the camera’s FOV were used. We evaluated three cases: LIDAR features only, image features only, and LIDAR and image features combined.

Figure 8 provides the threshold averaging ROC curves with respect to the three cases. As can be seen, the LIDAR

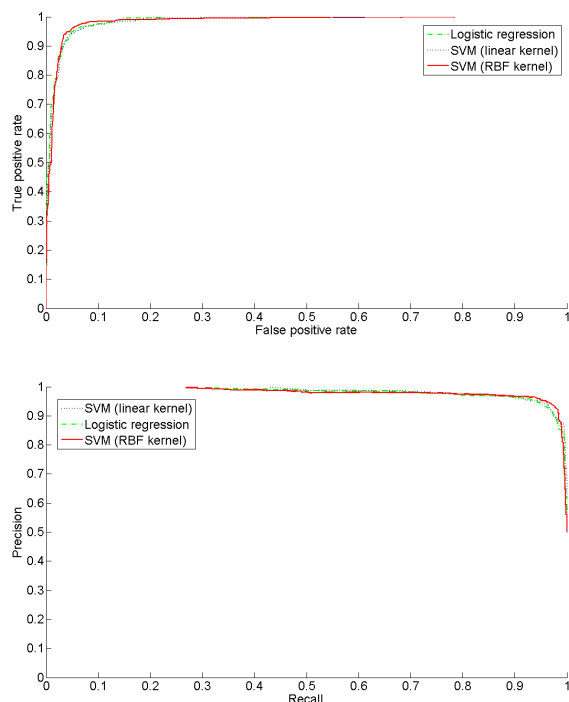


Fig. 5. The performance comparison of three different classifiers. ROC (top) and Precision/Recall (bottom) curves.

on this classification. Figure 4 shows an example of an automatically labeled scan, along with the corresponding imagery for reference.

Using this semi-automated method, we labeled data for training and testing. In total, we labeled 4400 instances – 1000 boundary and 1400 non-boundary instances from outside the camera’s field of view (FOV) (LIDAR data only) were used to compare the performance of three different classifiers and 800 boundary and 1200 non-boundary instances from within the camera’s FOV (image and LIDAR data) were utilized for the other tests. The labeled data was randomly split into three subsets: training (50%), holdout (25%), and test sets (25%). Each algorithm was trained using only the training data. The algorithms also had access to the holdout data to guard against overfitting. The testing data was kept sequestered. Once training is complete, each algorithm was evaluated on the testing data, and its performance was charted as a function of the classification decision threshold. This experiment was conducted 10 times with different randomly selected subsets of the available data. To measure the average performance of the experiment, the threshold averaging receiver operator characteristic (ROC) curves and precision-recall (PR) curves proposed by [14] were used. These two techniques are standard measures in the classification literature and differ only slightly in their definition. PR curves have the benefit that they are not as biased by data imbalance as ROC curves. Data imbalance occurs when there are significantly more examples of one type than the other [15].

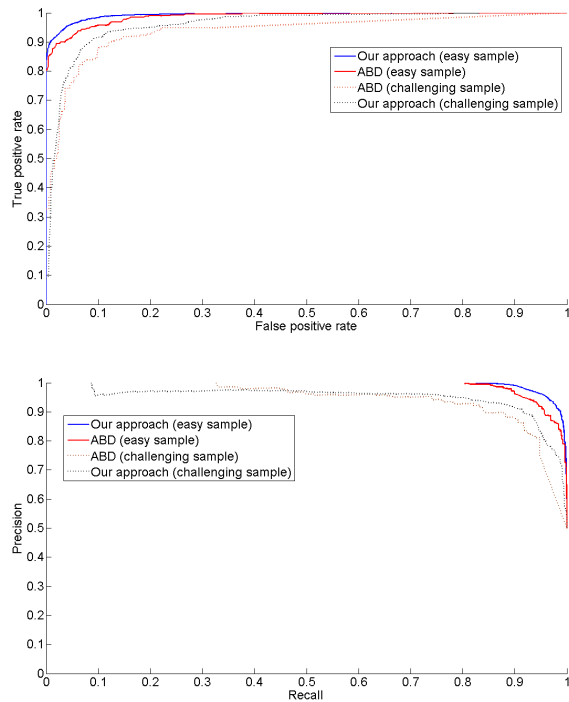


Fig. 6. Threshold averaging ROC (top) and Precision/Recall (bottom) curves. These illustrate that our method improves boundary detection performance when compared to the ABD algorithm.

features are more discriminative than the image-based features. This agrees with the intuition that depth discontinuities from 2D laser range measurements are a more reliable indicator of object boundaries than differences in image intensity or texture. However, the combination of LIDAR and image-based features performs better than features from either modality individually. This result suggests that the information in the LIDAR and image features is not entirely redundant.

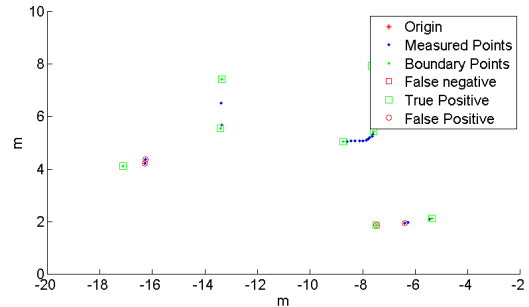
Figure 9 shows the segmentation result of LIDAR, image, and combined data when we consider a real frame data in the camera FOV. As can be seen in Figure 9a, the car on the left and the person standing 0.5 m in front are grouped into a single objects. Discriminating the objects to two different things is difficult because of their very close proximity to each other.

Figure 9b shows the segmentation result when only image features are only used for segmentation. In this experiment, the LIDAR data in the camera FOV are used. As can be seen, the segmentation result does not look so good. In the figure, the marked points indicate the manually labeled ground truth and each color box encompasses a set of points classified as non-boundary. In other words, the boxes represent the segmentation result. The height is fixed at 60 pixels.

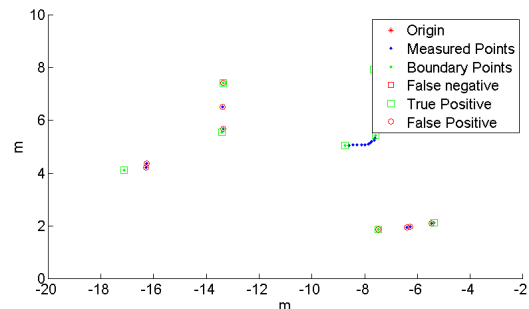
The method using combined LIDAR and imagery features was evaluated in the same frame of data (Figure 9c). In this case, boundaries that were mis-classified by the imagery feature method are now correctly classified. Specifically, the person and the car, which were separated by 0.5 m are now



(a) The scanned object



(b) Segmentation result (ABD)



(c) Segmentation result (our approach)

Fig. 7. An example showing the segmentation performance of the ABD and our algorithm for a single scan.

correctly divided into two segments. When only image or LIDAR features were used, these objects were segmented as one group.

## VI. SUMMARY AND FUTURE WORK

In this paper, we have proposed a supervised learning approach to object boundary detection. The method is aimed at line scanning LIDARs and can be augmented with imagery if available. We formulate the boundary detection task as a classification problem and use features derived from the LIDAR measurements and image patches surrounding the LIDAR measurements as inputs to a classifier. The classifier is learned using a set of labeled training examples. Experiments show that the SVM classifier performs slightly better than logistic regression. The proposed method improves on the performance of an existing dynamic thresholding boundary detection algorithm, especially for challenging boundary cases. Further improvement is achieved by incorporating visual features derived from co-registered imagery.

In future research, we hope to address a number of issues. First, we would like to evaluate additional types of image and LIDAR features for the boundary classification task. In particular, we are investigating features that are based

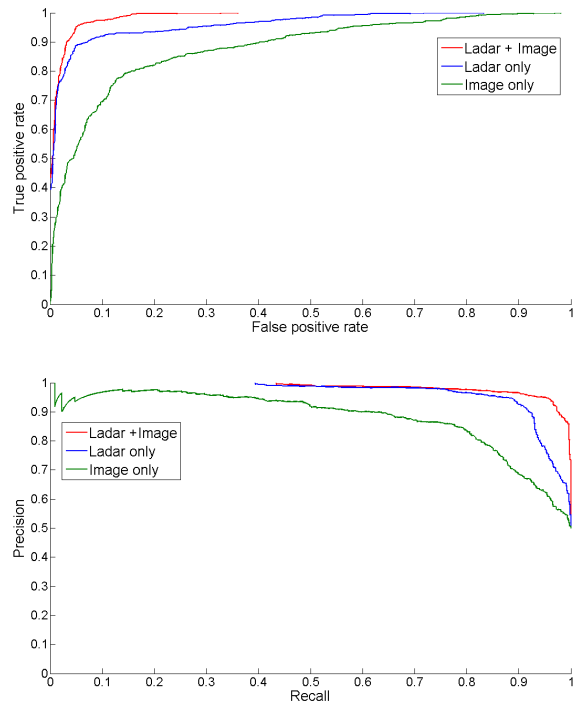
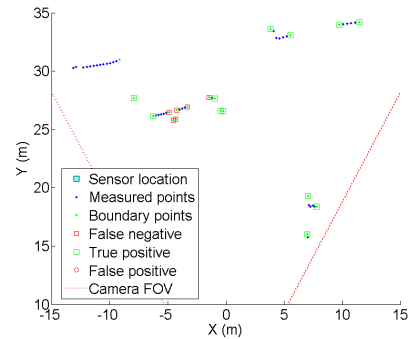


Fig. 8. Threshold averaging ROC (top) and Precision/Recall (bottom) curves when image only, LIDAR only, and combined features are used.

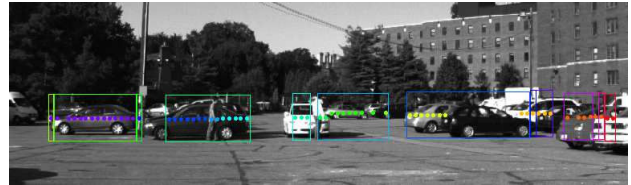
on image-based edge detection. Second, we plan to pursue alternate learning algorithms, such as AdaBoost [17], for the classifier. With larger numbers of features, the problem of feature selection becomes an issue, which can be addressed by methods like boosting. Third, we are working to integrate the boundary detection algorithm into a larger framework for detecting, modeling, and tracking moving objects using the combination of LIDAR and imagery.

#### REFERENCES

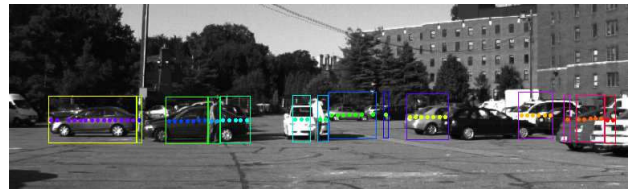
- [1] D.R. Martin, C.C. Fowlkes, and J. Malik, "Learning to detect natural image boundaries using local brightness, color, and texture cues," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 26, 2004, pp. 530-549.
- [2] M.A. Ruzon and C. Tomasi, "Color edge detection with the compass operator," *Computer Vision and Pattern Recognition, IEEE Computer Society Conference on.*, 1999.
- [3] M. Maire, P. Arbelaez, C. Fowlkes, and J. Malik, "Using contours to detect and localize junctions in natural images," *IEEE Conference on Computer Vision and Pattern Recognition. CVPR 2008*, 2008, pp. 1-8.
- [4] A. Stein and M. Hebert, "Occlusion boundaries from motion: low-level detection and mid-level reasoning," *International Journal of Computer Vision*, vol. 82, May. 2009, pp. 325-357.
- [5] A.N. Stein and M. Hebert, "Local detection of occlusion boundaries in video," *Image and Vision Computing*, vol. 27, 2009, pp. 514-522.
- [6] G.A. Borges and M. Aldon, "Line extraction in 2D range images for mobile robotics," *J. Intell. Robotics System*, vol. 40, 2004, pp. 267-297.
- [7] C. Prenebida and U. Nunes, "Segmentation and geometric primitives extraction from 2d laser range data for mobile robot applications," *Robotica*, 2005, pp. 17-25.
- [8] K. Arras, O. Mozos, and W. Burgard, "Using boosted features for the detection of people in 2D range data," *Robotics and Automation, IEEE International Conference on*, 2007, pp. 3402-3407.
- [9] L. Spinello and R. Siegwart, "Human detection using multimodal and multidimensional features," *Robotics and Automation, IEEE International Conference on*, 2008, pp. 3264-3269.



(a) The segmentation result of a single scan LIDAR data.



(b) The segmentation result of a single frame image data.



(c) The segmentation result based on image and LIDAR features

Fig. 9. The boxes represent each cluster. The distance between the person and the car is only 0.5m and the sensors' origin is placed at 25m from the objects.

- [10] D.W. Hosmer and S. Lemeshow, *Applied logistic regression*, Wiley-Interscience, 2004.
- [11] V.N. Vapnik, *The nature of statistical learning theory*, Springer Verlag, 2000.
- [12] M.J. Swain and D.H. Ballard, "Color indexing," *International Journal of Computer Vision*, vol. 7, 1991, pp. 11-32.
- [13] Camera calibration toolbox for Matlab. [Online]. Available: [http://www.vision.caltech.edu/bouguetj/calib\\_doc/](http://www.vision.caltech.edu/bouguetj/calib_doc/)
- [14] T. Fawcett, "An introduction to ROC analysis," *Pattern recognition letters*, vol. 27, 2006, pp. 861-874.
- [15] J. Davis and M. Goadrich, "The relationship between precision-recall and ROC curves," *Proceedings of the 23rd international conference on Machine learning*, Pittsburgh, Pennsylvania: ACM, 2006, pp. 233-240.
- [16] C.C. Chang and C.J. Lin, *LIBSVM: a library for support vector machines*, 2001.
- [17] R. Schapire and Y. Freund, "A decision theoretic generalization of on-line learning and an application to boosting," *Journal of Computer and System Sciences*, vol. 55, 1997, pp. 119-139.