Pedestrian Detection Using Correlated Lidar and Image Data

Samuel Rohrer
University of Michigan
rohrer@umich.edu

Ian Lin
University of Michigan
tiannis@umich.edu

Abstract

Recent progress in software and hardware systems will lead to an increase in daily interactions with robots and intelligent systems. As a result of this safety must be at the forefront of the design of autonomous systems, particularly autonomous cars. With this in mind we aim to create a pedestrian detection system based off of sensors commonly found on autonomous vehicles. We will use correlated Lidar and image data along with algorithms found in recent papers to perform pedestrian detection.

1. Introduction

As progress in software and hardware continues, it is expected that autonomous robotics and intelligent systems will begin to play an integral role in everyday life. Jobs that are physically demanding or unsafe can be replaced by robots and autonomous cars can greatly improve road safety and efficiency. In order for these intelligent systems to succeed, safety must be at the forefront of their design. Detection of obstacles is a very significant aspect of this concern, especially for autonomous cars which must detect not only other cars, but also road signs, traffic signals, and pedestrians. In particular, pedestrian detection is an ongoing research area within computer vision.

The goal of this project is to detect pedestrians in a preexisting dataset: the LIPD dataset provides correlated LIDAR data and image data. We aim to do this by identifying obstacles on the road, based on LIDAR data, and associating these obstacles to the image data through feature extraction. After feature extraction, we plan to use a trained classifier to determine if the obstacle is indeed a pedestrian in the road. We could extend this by training a more sophisticated classifier that has the ability to discern the difference between cars, pedestrians, and road signs.

2. Previous Work

There has been a good deal of work in the area of pedestrian detection for mobile robotics. In most systems, the sensor subsystems operate independently of each other. In [4] the authors use Lidar to generate regions of interest, then use image analysis to determine the presence of a pedestrian. Some other sensors available on modern autonomous robotics platforms include: radar, ultrasound, camera sensors and Lidar (light detection and ranging). However, each of these sensors comes with their own inherent problems. Cameras can fail in low light situations, and Lidar can fail when objects are side by side at the same distance. Both can fail in poor weather, as Lidar can fail if rain or snow causes the light to be reflected too early and camera images can fail if the lens is obstructed by poor weather.

In order to complete this project, we started with a data set [1]. Some benefits of this dataset include: correlated Lidar and image data, and a classification dataset. The correlated datasets were composed of a forward facing camera (limited field of view relative to Lidar) and four Lidar scans all facing outwards at different angles. The classification dataset had both training and testing sets for the Lidar.

3. Technical Work

We plan to use both the Lidar data and image data together to identify pedestrians on the road. With Lidar data, we can identify obstacles on the road and then analyze the corresponding images that potentially contain pedestrians. There has already been work done in the areas of correlating Lidar and single image data for the purpose of detecting pedestrians. Most of it follows the same approach that we are outlining and has had succes. For Lidar object detection, we decided to follow the pipeline outlined in [4], and add stages for error correction and correlation with image data. The pipeline is as follows: segmentation, feature extraction, classification, error correction, and correlation with image data.

3.1. Lidar Segmentation

The pipeline begins with segmentation of the Lidar data to find the points where the directionality of the segment changes. According to [5] if it is a new segment then (1) holds true where r_i is the distance to current point, r_{i+1} is the distance to next point, $cos(\Delta\alpha)$ is the 3D angle between the two points, and C_0 is a constant to adjust for noise:

$$\sqrt{r_i^2 + r_{i+1}^2 - 2r_i r_{i+1} cos(\Delta\alpha)} > thd \quad (1)$$

$$thd = C_0 + \sqrt{2(1 - cos(\Delta\alpha} * min(r_i, r_{i+1}) \quad (2)$$

For the purpose of this project any segment less than three points long was discarded before moving to feature extraction. This was performed for each Lidar scan in isolation of the other scans.

3.2. Feature Extraction

After separating the Lidar scan data into different segments we then performed feature extraction on the segments. We used 10 different features for each segment, based on [4].

Feature num	Formula	Description
1	np	number range points
2	$np * r_{min}$	number of range points * minimum range distance
3	$\sqrt{\Delta X^2 + \Delta Y^2}$	RMS of segment width and height
4	$\frac{1}{np}\sum_{n=1}^{np} \|\|x_n - x_m\|\|$	Mean average deviation from median (x_m)
5	$\frac{1}{np}\sum_{n=1}^{np}(x_n - x_{lsq})^2$	Linearity - distance to the least squares line (x_{lsq})
6	$\frac{1}{np}\sum_{n=1}^{np}(x_n - \mu_x)^2$	Second central moment (about mean μ_x)
7	$\frac{1}{np}\sum_{n=1}^{np}(x_n - \mu_x)^3$	Third central moment (about mean μ_x)
8	$\frac{1}{np}\sum_{n=1}^{np}(x_n - \mu_x)^4$	Fourth central moment (about mean μ_x)
9	$\sum_{n=1}^{np} \|\|x_n - x_{n-1}\|\|$	Segment length (norm distance between points)
10	$\sigma(ft9)$	Standard deviation of segment length (norm distance between points)

Figure 1. Initial segmentation results. Green squares are identified segments. Red, blue, yellow and green dots are Lidar points.

These features describe the distribution of segment points and were all able to be computed quickly and robustly for all segments greater than three points long.

3.3. Classification

For classification we used training data from [1] to train a Python SkLearn DecisionTreeClassifier [3]. The training data consisted of pre-segmented lidar scans that were very clean and well segmented. The lidar scans were split between segments with pedestrians and segments without pedestrians. A similar set of test data was also provided and our classifier achieved an accuracy of 88%.

4. Experimental Results

We ran our pipeline on raw lidar scans correlated with image data. After segmenting each of the 4 lidar scans individually, we pass each segment to the feature extractor. The feature extractor outputs a feature vector composed of the 10 features described above and our classifier is able to make a binary classification of whether or not the segment contains a pedestrian. To visually display our results, we overlay each of the four lidar scans over its corresponding image. We color each lidar scan a different color–yellow, red, green, and blue– to distinguish between them. Since the lidar scan is wider than the image, we do not display lidar points beyond the width of the image. Segments that are positively classified as pedestrians are saved. Their start and end are plotted as large green squares.

4.1. Example Lidar Output

An example of the output of our system can be seen in Figure 1. Our lidar classifier does pick up the two woman

Figure 2. Example output of OpenCV's peopledetect [2].

Figure 3. Example output of potential system that combines lidar with peopledetect. Bounding boxes are handdrawn.

on the right side of the image as two distinct pedestrians. There are additional people behind these two woman who are largely occluded and are not identified by our classifier. Unfortunately we also get several false positives around the cars on the left side of the road.

4.2. Example OpenCV Output

An example of the output of OpenCV's peopledetect [2] code can be seen in Figure 2. The results are surprisngly similar to our lidar classifier: it picks up the two women as well as two extra false positives. The output of OpenCV is more clearly indicated than the output of our lidar scans as they can draw very nice bounding boxes around the pedestrians. Unfortunately lidar scans do not provide any height information that can be extrapolated so estimated bounding boxes would have to be drawn with some assumption regarding the average height of a pedestrian.

5. Future Work

As seen in Figure 1 and mentioned above, our lidar classification does output a large number of false positives. While this is somewhat better than false negatives– identifying too many pedestrians is preferable over missing one–we believe that our results could be improved by minimizing false positives and more closely tying image classifications with our lidar classifications.

5.1. Minimizing False Positives

Since our lidar data consists of four horizontal scans taken at slightly different angles, we should expect that the existance of pedestrians results in several positive segment classifications (up to four) in the same area. Therefore, a simple way to decrease the number of positive classifications is to only positively classify segments if there are at least two other segment classifications in agreement within a small pixel margin. Taking Figure 1 for example, we get a large number of segments on the right side of the image around the two women which would be saved since there are many segments in agreement vertically. However, the other false positives would mostly be ignored since several are mostly isolated (although the white car does appear to have 3 segments in agreement as well).

5.2. Correlation with Image Data

Another addition that could improve our results is making a final prediction based on the results of both the lidar classification and image classification. A simple set intersection of the general areas where the positive lidar segments and OpenCV bounding boxes are in agreement could cut down on the extra classifications coming from both our lidar classifier as well as OpenCV's peopledetect. An example of a potential output of such a system can be seen in Figure 3.

6. Conclusion

We were able to successfully segment raw lidar scans, extract features, and make a classification. Unfortunately the final dataset that we test our system on provides labels as a series of bounding boxes on the image itself. As such, we did not have labels for individual segments and do not have an exact accuracy for our classifier. However, our intial examination of our results does look fairly positive: we generally do pick up pedestrians but also pick up other objects on the side of the road such as cars.

Addition of the OpenCV peopledetect code provides additional interesting results. Similarly to our lidar classifier, OpenCV does generally detect pedestrians but also provides incorrect bounding boxes around cars and buildings. These incorrect classifications generally do not coincide with our lidar classifications and thereby validates our intial beliefs that a system combining both Lidar and Image classifications can perform better than either individually.

References

[1] Laser and image pedestrian detection dataset. `http://home.isr.uc.pt/~cpremebida/dataset`.

[2] Open source computer vision. `http://opencv.org`.

[3] Python scikit-learn. `http://scikit-learn.org/stable/index.html`.

[4] Carlos Premebida, Oswaldo Ludwig, and Urbano Nunes. Lidar and vision-based pedestrian detection system. *Journal of Field Robotics*, 26(9):696–711, 2009.

[5] C. Premebida and U. Nunes. Segmentation and geometric primitives extraction from 2d laser range data for mobile robot applications. In *Proceedings of the 5th National Festival of Robotics Scientific Meeting*, pages 17–25, Coimbra, Portugal, 2005. ROBOTICA.