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 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 determine which features corresponded to a pedestrian. To implement this classifier we used Python's SkLearn DecisionTreeClassifier [3]. It was trained on the labeled pedestrian segments from the training set. On very clean, well segmented data (also available from [4]) our classifier has an accuracy of 88

3.4. Error Correction

Our initial results were somewhat prone to error, as seen in Figure 1.

To account for this error, we decided to only classify a pedestrian from the Lidar data if we found at least three identifications across Lidar scans (out of four Lidar scans) within a 50 pixel margin. Results of this will be discussed in the Experimental Results section.

3.5. Correlation with Image Data

In order to create a classifier even better than the one proposed in [4] we decided to add another check before positively identifying a pedestrian. To do this we modified the peopledetect.py sample file from OpenCV [2] to return the bounding boxes of all pedestrians instead of just drawing them. Then if the Lidar classifier with error correction and the OpenCV classifier both classified a pedestrian we said that a pedestrian was in fact present in the image.

4. Experimental Results

5. Conclusion

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] O. L. Carlos Premebida and U. 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.