Pedestrian Detection and mapping

Assaf Livne & Oded Yechiel

Advance Computer Vision, 361-2-2030

Department of Electrical Engineering

BGU, 2015

**Abstract** – Pedestrian detection is a key problem in computer vision and is a key element in vision systems for autonomous vehicles. For autonomous vehicles it is a very important task the main reason is safety. The pedestrian detection algorithms were explored widely and thoroughly by a wide range of explorers. Perhaps the best algorithm that can be found today in the pedestrian detection community is the algorithms and open code written by Piotr Dollar, which we are going to use in our work. However, detecting the pedestrians is just one piece of the puzzle, and the other piece is to map the pedestrian in the three dimensional world. In this paper we will show our results of working with Dollar’s algorithm and our implementation for detecting and mapping.

# Introduction

Autonomous vehicles and robots require a vision system and mapping capabilities. Whether this vision system and mapping is performed using LiDAR, cameras, or other sensors the vision and mapping is mandatory for the system to operate. Although LiDAR sensors are accurate robust and easy to use, they have some drawbacks such as high market prices and especially the problem of extracting features of the points themselves. Cameras, on the other hand, are very cheap and mimics the human eye in the sense of getting the scenery input. However, extracting meaningful data from the images is extremely complicated, and 3D reconstructions of the scenery is nearly impossible without any prior information.

It is desired to know the where about of the surrounding pedestrians for obvious reasons. There are numerous algorithms and methods [1] that can detect pedestrians such as VJ, Shapelet, and ConvNet that uses different types of features such as Haar, Gradients and HOG. Each method has its pros and cons in the sense of computation time and accuracy. For the time being, Piotr Dollar’s is probably the best pedestrian detector algorithm for real time application due to its fast computation and accurate [2].

Indeed, the algorithm suggested by Piotr Dollar does not solve the solution completely, but gives only one piece of the puzzle. The algorithm finds the pedestrians in a given image, but still the mapping of the person into a 3D space is still an issue.

In this project we would like to try to detect and map pedestrians in a 3D space while using and improving Dollar’s algorithm. The improvement is performed by using a tracking algorithm, e.g. Kalman filter.

# Pedestrian Detector

.

Dollar's FPDW (the Fastest Pedestrian Detector in the west) is faster than all other documented methods. To achieve a real time system that can identify and track pedestrians we need to build a fast algorithm and for the part of the identifying the FPDW is the fasts. Dallar's work is good not only because of the fast algorithm but also for the accuracy of the detection. As you can see in Figure 1 the FPDW has the fastest results at 1 false positive per image on 640 X 480 images and have a detection rate less only by few percent compare to other methods. The FPDW is based on HOG detector with Dollar's technique that can approximate an image pyramid. This saves time by avoiding constructing an image pyramid over and over again for each group of pixels. In Figure 2 we can find the results of the FPDW on two different datasets. We can see that by increasing the "miss rate" we will decrease the "false positive per image rate" so increasing the threshold, meaning low false positive rate, will give us higher miss rate. This is the main con of the FPDW (and all other pedestrian detectors today). To track and identify pedestrians on a video feed you can use this method with a fix threshold but the result will not be satisfying. If your threshold will be high you will miss most of the pedestrian on the frames, for low threshold you may identify all of the pedestrians but you will also get a lot of false positive along with it. That is why we decided to work with high threshold and to use other tools to fill the empty spots.

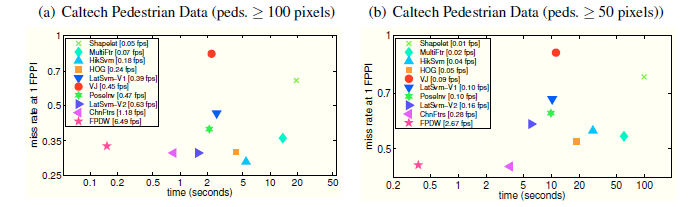


Figure - Time versus detection rate at 1 false positive per image on 640 X 480 images from Caltech Pedestrian Dataset. FPDW obtains a speedup of about 10-100 compared to competing methods with a detection rate within a few percent of best reported performance.

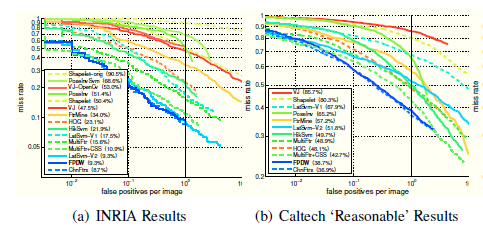


Figure - miss rate versus false positives per image in 640 X 480 images from INRIA Dataset (a) and Caltech Pedestrian Dataset (b). In both of the cases the detection rate of FPDW is within a few percent of ChnFtrs while being 1-2 orders of magnitude faster that all competing methods.

# Improvement of the Pedestrian Detector

The pedestrian detector algorithm proposed by Dollar is still not good enough to use as is due to its high false positives when using a low threshold or non-detect when using a high threshold. The main idea for now is to use a high threshold and to use a Kalman filter on the image to try and track the pedestrian between non-detection. The Kalman filter model is used as follows:

( 1 )

( 2 )

Where F is the state space model and B is the input vector, which in our case we use a constant acceleration input of. The measurement vector,, consist of the X and Y pixel of the center of the square, i.e.,

( 3 )

And thus the matrix is as follows:

( 4 )

Using a standard and constant process noise covariance,

( 5 )

And a small measurement noise of 5 pixels,

( 6 )

Thus, the Kalman filter estimation is as follows:

Predict stage:

( 7 )

( 8 )

Update stage:

( 9 )

( 10 )

( 11 )

When doing so, once the pedestrian is detected at least twice, which is quite probable when working at a high frame rate, the Kalman filter will continue and detect the pedestrian. The upside of this form of algorithm is that we were able to improve the detection and tracking of pedestrians from about 60% at a threshold of 150 to 90% detection rate when the camera is static. Although the great detection rate, this improvement has 2 big disadvantages. The first is that it is hard to track how many pedestrians are in the data base and which new measurement belongs to which pedestrian in the database. This issue might be solvable on an algorithmic level. The second flaw of this algorithm is that it doesn’t evaluate the movement of the vehicle. The model state F and B is the state space model of the pedestrian, but the movement of the cameras is not taken into account. Even more, the movement of the cameras are known due to the existence of a GPS and IMU systems installed alongside the cameras.

This 2 issues described above got us to a conclusion that it is better to try attacking the problem from a different angle which is stated below.

# Moving from 2D to 3D

Due to the problems described in section 3, it sounds reasonable to track the pedestrians after mapping them to the 3D world (see Figure 3). In this manner the Kalman Filter tracking and estimation may be performed using also the GPS/IMU sensor installed on the vehicle.

FPDW

Map pedestrian in 3D world

Kalman

Filter

tracking

Figure – Algorithm for mapping pedestrians on a 3D map.

The main question now is how to perform the 2nd level in the algorithm? How to map the pedestrians to the 3D map (Note that the map is actually 2D – we have no interest in the height axes).

To get the 3D location of the pedestrian we have 2 options. The first is by detecting the pedestrian in 2 cameras, and use stereo theory in order to triangulate the location of the pedestrian in 3D space. This might sound easy in theory, but it has major complications in real life application. The second option is to use a monocular system with priory known values in order to get the distance. In this option what we done is to take a large number of images with a pedestrian at known distances. Run the FPDW algorithm to detect the pedestrian and to cross the distance with the size of the rectangle (see Figure 4). When this is done for all measurement a function is built to estimate the distance as a function of the size of the rectangle (see Figure 5).



Figure – Calculate the size of the rectangle as a function a known distance.

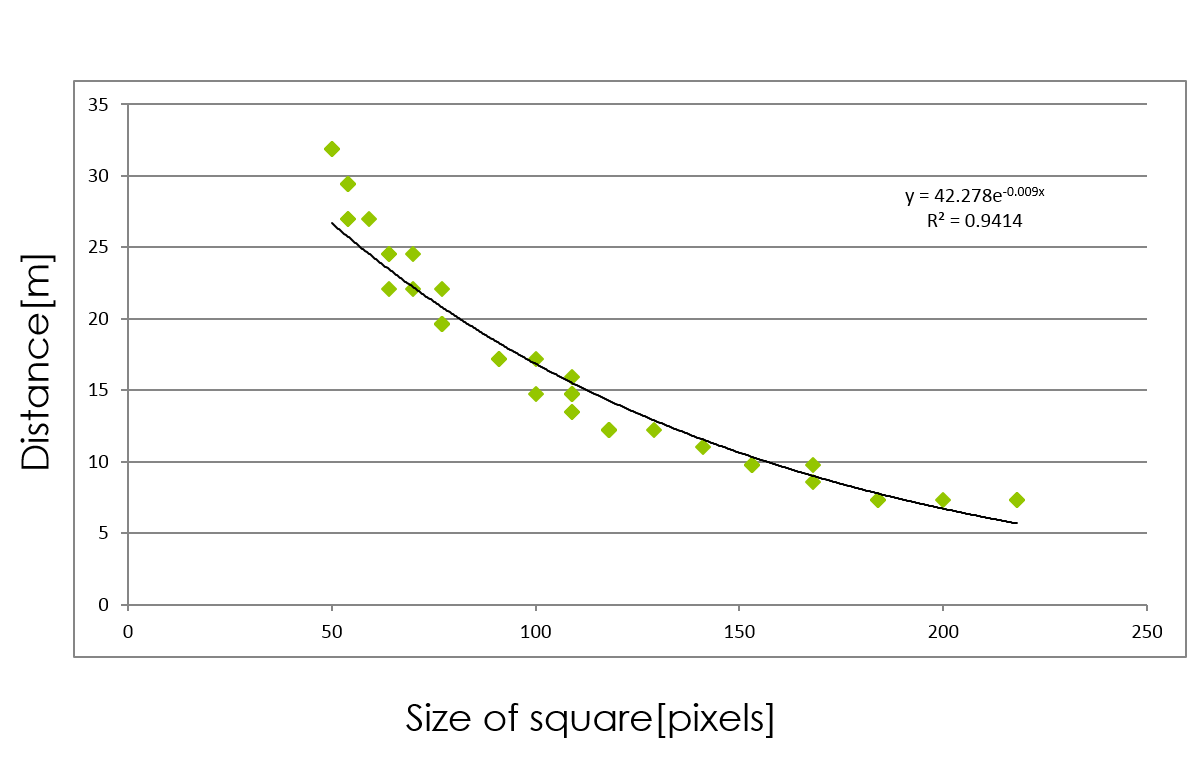


Figure – the measurement VS the size of the square.

Now that we have the function, we can estimate the distance of the pedestrian using the following formula

( 12 )

Where z is the depth, and w is the size in pixel of the rectangle.

Now that we have depth, z, we can find x by using the simple pinhole model

( 13 )

By doing a simple calibration we can obtain, and thus achieve x and y.

# Kalman filter on the 3D data

Now that we have this 3D data we can move on forward and track the pedestrians as stated in Figure 3. However, as opposed to the model described in section 3 the model in the Kalman filter the model (equation (7)) now will be,

( 14 )

Where is the location of the car at time k.

# Results

Due to limited testing environment it is quite difficult to output exact results of the algorithm. However, for distances of approximately 8-20 [m] the mapping and tracking of the pedestrians seems acceptable, however, for close ranges the algorithm results is not satisfactory. More tests need to be done.

# Conclusion

Dollar algorithm is a good starting point for mapping the pedestrians in the environment on a 3D world map, however, the task doesn’t stop there. In order to complete this task other techniques and algorithms are required. Using predefined function gives us results that are bearable but still require more adjustment to improve the mapping. Adding a Kalman filter to track the pedestrian has better output when this is done in the 3D space instead on the image 2D space when the camera is dynamic.

# Future work

Comparing the results to ground truth is mandatory and will be performed using a Velodyne LiDAR system. In addition, moving to Stereo vision to map the pedestrians might result in better performance and will be explored in later research.

# References

|  |  |
| --- | --- |
| [1] | R. Benenson, M. Omran, J. Hosang and B. Schiele, Ten years of pedestrian detection, What have we learned?, Saarbrücken, Germany, 2014. |
| [2] | P. Dollar, S. Belongie and P. Perona, The Fastest Pedestrian Detector in the West, Aberystwyth, UK, 2010. |
| [3] | A. Ess, B. Leibe, K. Schindler and L. van Gool, A mobile vision system for robust multi-person tracking, 2008. |
| [4] | P. Dollar, B. Badenko, S. Belongie, P. Perona and Z. Tu, Multiple component learning for object detection, 2008. |
| [5] | C. Schiele and B. Wojek, A performance evaluation of single and multi-feature people detection, 2008. |