WestVirginiaUniversity
THE RESEARCH REPOSITORY @ WVU

Graduate Theses, Dissertations, and Problem Reports

2017

Point Cloud Processing Algorithms for Environment Understanding in Intelligent Vehicle Applications

Ahmed Cheikh Sidiya

Follow this and additional works at: https://researchrepository.wvu.edu/etd

Recommended Citation
Cheikh Sidiya, Ahmed, "Point Cloud Processing Algorithms for Environment Understanding in Intelligent Vehicle Applications" (2017). *Graduate Theses, Dissertations, and Problem Reports*. 5340.
https://researchrepository.wvu.edu/etd/5340

This Thesis is protected by copyright and/or related rights. It has been brought to you by the The Research Repository @ WVU with permission from the rights-holder(s). You are free to use this Thesis in any way that is permitted by the copyright and related rights legislation that applies to your use. For other uses you must obtain permission from the rights-holder(s) directly, unless additional rights are indicated by a Creative Commons license in the record and/ or on the work itself. This Thesis has been accepted for inclusion in WVU Graduate Theses, Dissertations, and Problem Reports collection by an authorized administrator of The Research Repository @ WVU. For more information, please contact researchrepository@mail.wvu.edu.

Point Cloud Processing Algorithms for Environment Understanding in Intelligent Vehicle Applications

Ahmed Cheikh Sidiya

Thesis submitted to the
Benjamin M. Statler College of Engineering and Mineral Resources
at West Virginia University
in partial fulfillment of the requirements
for the degree of

Master of Science
in
Computer Science

Xin Li, Ph.D., Chair
Yaser P. Fallah, Ph.D.
Victor Fragoso , Ph.D.

Lane Department of Computer Science and Electrical Engineering

Morgantown, West Virginia
2017

Keywords: Convolutional Neural Network, Machine Learning, Point cloud data , Autonomous driving, sensors fusion

Copyright 2017 Ahmed Cheikh Sidiya

Abstract

Point Cloud Processing Algorithms for Environment Understanding in Intelligent Vehicle Applications

Ahmed Cheikh Sidiya

Understanding the surrounding environment including both still and moving objects is crucial to the design and optimization of intelligent vehicles. In particular, acquiring the knowledge about the vehicle environment could facilitate reliable detection of moving objects for the purpose of avoiding collisions. In this thesis, we focus on developing point cloud processing algorithms to support intelligent vehicle applications. The contributions of this thesis are three-fold.

First, inspired by the analogy between point cloud and video data, we propose to formulate a problem of reconstructing the vehicle environment (e.g., terrains and buildings) from a sequence of point cloud sets. Built upon existing point cloud registration tool such as iterated closest point (ICP), we have developed an expectation-maximization (EM)-like technique that can automatically mosaic multiple point cloud sets into a larger one characterizing the still environment surrounding the vehicle.

Second, we propose to utilize the color information (from color images captured by the RGB camera) as a supplementary source to the three-dimensional point cloud data. Such joint color and depth representation has the potential of better characterizing the surrounding environment of a vehicle. Based on the novel joint RGBD representation, we propose training a convolution neural network on color images and depth maps generated from the point cloud data.

Finally, we explore a sensor fusion method that combines the results given by a Lidar based detection algorithm and vehicle to everything (V2X) communicated data. Since Lidar and V2X respectively characterize the environmental information from complementary sources, we propose to get a better localization of the surrounding vehicles by a linear sensor fusion method. The effectiveness of the proposed sensor fusion method is verified by comparing detection error profiles.

Acknowledgments

It is a pleasure for me to express my sincere gratitude to Dr. Yaser P. Fallah, for giving me the opportunity to work under his supervision, for his patience and guidance. I would also like to thank Dr. Xin Li, for his continued guidance, his words of encouragement. His advice and mentorship helped me in many ways.

I thank my family for their encouragement and support throughout my studies.

Contents

Abstract **ii**

Acknowledgments **iii**

List of Figures **vi**

1 Introduction **1**
- 1.1 Literature Review 2
- 1.2 Contribution 3
- 1.3 Organization of Thesis 3

2 Environment Reconstruction and understanding **5**
- 2.1 Constructing the vehicle environment from multiple point clouds 5
  - 2.1.1 Simultaneous Alignment of Multiple Point Clouds:Hierarchical Extension of ICP 6
  - 2.1.2 Outlier Rejection by Robust EM-ICP 7
  - 2.1.3 Segmentation of the background 8
- 2.2 Constructing depth maps from the point cloud data 8
  - 2.2.1 Depth maps creation 9
  - 2.2.2 Using convolution neural network on depth map for the purpose of pedestrian detection 13
    - a- Generating training and testing data from KITTI dataset 14
    - b-CNN architecture 15
    - c-CNN hyperparameters 16

3 Fusion of Information from Lidar sensor and V2X Communicated Data for Automated Driving **17**
- 3.1 Proposed Framework 19
- 3.2 fusion algorithm 20
- 3.3 Fusion method 21
- 3.4 Detection Error Profile 21
  - 3.4.1 Lidar method error profile 21
  - 3.4.2 GPS error profile 31
  - 3.4.3 Defining the weights 32

4 Results and Analysis 33
4.1 EM-ICP algorithm 33
4.2 Convolution neural network on depth maps 36
4.3 Lidar and GPS data fusion 37

5 Conclusion 38

References 39

List of Figures

2.1 Two successive unaligned point clouds, Pk and Pk+1 6
2.2 The distribution of matching errors in ICP algorithm as a function of displacement (outliers are declared by those a certain threshold for example .12m). 8
2.3 Segmentation of still background objects in a complex scene (different objects are highlighted by different colors). 9
2.4 RGB and point cloud data 10
2.5 Sparse I map and final depth map D 10
2.6 Depth generated respectively with the y and z components 11
2.7 the RGB data and the XYZRGB data for comparison 12
2.8 Results after combining multiple XYZRGB data to create the vehicle environment 13
2.9 Positive samples 14
2.10 Hard negative samples 14
2.11 Medium and Easy negative samples respectively 15
2.12 Graphical representation of CNN using a Caffe[1] tool 16

3.1 Fusion Framework 19
3.2 A rough sketch of how the different methods compare 20
3.3 Examples of the point cloud data from LIDAR, as well as a snapshot of a scene and detected vehicles are shown at the top 22
3.4 Scatter plot of X predicted and X real 23
3.5 Scatter plot of Z predicted and Z real 24
3.6 Histogram of the error 25
3.7 Histogram of the error values between 0 and 1 26
3.8 PDF of the error values between 0 and 1 27
3.9 Histogram of error values superior to 1m 27
3.10 table containing the 95 percentile for all error values 28
3.11 plot of the 95 percentile for all error values 28
3.12 table containing the 95 percentile for error values between 0 and 1m 29
3.13 plot of the 95 percentile for error values between 0 and 1m 29
3.14 Error profile of Lidar based method 30

4.1 Urban environment reconstruction from Ford and KITTI data 34
4.2 Road intersection plus rural environments reconstruction from KITTI data . 34

4.3 Example of Extracted Foreground 35
4.4 X component translation and vehicle speed 35
4.5 Graph of the results of training the CNN 36
4.6 Graph of the results of the fusion algorithm - σ_L from experiment and σ_G from simulation 37

Chapter 1

Introduction

Obtaining information about the surrounding environment is critical to the applications related to intelligent vehicles. Environment-related information can help vehicles drive more safely (e.g., avoid collisions) and efficiently (e.g., route planning). Various sensors including visible-spectrum cameras, radars and LIDARs have provided a rich collection of data that can be exploited by intelligent vehicles; though the cost and accuracy of different sensors (e.g.,camera vs. LIDAR) could vary [1], [2]. Among those sensors, LIDAR-based data have received increasingly more attention recently due to rapid advances in both hardware (e.g., from Velodyne HDL-16e, 32e to 64e) and software (i.e., point cloud registration [3],[4],[5], segmentation [6] and classification [7]). Under the context of intelligent vehicles, LIDAR-based approaches have been studied for the detection/tracking of pedestrian [8], [9] curb [10],[11] and vehicle [12],[7].

The motivation behind this work is largely two-fold. On one hand, even though LIDAR-based environment reconstruction was studied for online mobile mapping in [13], the issue of outliers (e.g., moving objects should not contribute to the reconstruction of a still environment) has not been addressed yet. Therefore, it is desirable to develop a robust technique capable of separating moving objects from the still background. Meanwhile, there is a need to distinguish regular from irregular (e.g., jaywalking of a pedestrian, sudden lane change of a vehicle) events. The latter – to the best of our knowledge – has not been considered in the open literature of intelligent vehicles.

On the other hand, how to fuse data from multiple sources (e.g., LIDAR sensor vs. vehicle-to-everything communication) is a problem that has been underresearched in the literature. LIDAR sensors capture local information but their capabilities are constrained by the physical limit of those sensors (e.g., operating range); vehicle-to-everything (V2X) communication often supplies some supplementary information about the environment through the vehicle communication system. Integrating these two kinds of data could help an intelligent vehicle better understand the surrounding environment and make more informed decisions.

For environment reconstruction, we propose to develop a robust point cloud mosaicking technique combining iterative closest point (ICP) based on registration [14] and EM-based background estimation [15]. Our approach is similar to the work [5] of surface registration but ours is specifically tailored for point cloud data and has been tested on large-scale data set. Furthermore, we have trained a convolution neural network on depth maps obtained by projecting the 3D points onto an image plane and applying filtering method. Our work targets at exploiting the supplementary color information for better reconstruction and understanding of the vehicle environment.

For information fusion, we propose a new fusion algorithm for fusing the data generated by lidar detection algorithm with V2X Communicated Data. The proposed linear fusion method works with the position (distance) data from LIDAR and GPS data over Dedicated Short Range Communications(DSRC). A statistically optimal fusion strategy based on detection error profile was developed and experimentally verified using real-world data. Our experimental results have shown *[Ahmed, you need to provide a summary of your experimental results from Sec. 4.3 here]*

1.1 Literature Review

When compared against image data, LIDAR admits computationally more efficient processing and enjoys less vulnerability to adversary weather and lighting conditions. Therefore, point cloud processing has received increasingly more attention in recent years. Registration of 3D data originated from the seminal work of iterative closest point (ICP) [14] developed in 1990s. Since then, many variants of ICP techniques have been developed and applied to various data sets (e.g., [4], [5], [13]). However, most existing works assume that the data sets are clean and complete. The issue of outlier detection robustness has not been addressed in the open literature as far as we know. A related problem is the reconstruction of surface from point cloud data [16]. This is a classical problem that has been extensively studied by computer graphics community.

Just like image data, clustering and segmentation of point cloud data are handy tools supporting other high-level tasks such as classification and recognition. Existing works (e.g., [6],[17],[7],[18]) have mostly focused on the use of geometric features/clues for clustering and segmentation. With the separation of moving objects from still background (vehicle environment), we argue it is more fruitful to exploit motion-related information (e.g., trajectory, speed and acceleration).

However, under the context of intelligent vehicles, efficient reconstruction and visualization of point cloud data often faces new challenges due to their large size and arising from the desire of exploiting structural a priori information (e.g., urban vs. rural environment).

Several techniques have been proposed in the literature for the purpose of object recognition and detection in 3d environment. Just like in the case of images, the features used can be manually engineered [19][17] or learned using a convolution neural network. Neural network techniques are divided into two categories; one that uses 2D depth maps generated from point cloud [20] and others that leverage the techniques of 3D convolution [21] [22] to deal with the 3d nature of the data.

A lot of work has been carried in the topic of sensor fusion algorithms by researcher, R.E. Kalman published his famous paper proposing a recursive solution to the discrete-data linear filtering problem [23], [24] presents a fusion method that is based on Bayesian network, and [25] solution is based on central limit theorem.

1.2 Contribution

The main contributions of this work can be stated as follows:

1. A robust point cloud mosaicking technique has been proposed combining iterative closest point (ICP) based registration and EM-based background estimation.

2. A convolution neural network based detection of pedestrian in depth maps.

3. Adding color information to the vehicle environment

5. A sensor fusion method between local and V2X communicated data.

1.3 Organization of Thesis

This experiment can be organized in two different parts. The first part describes the vehicle environment reconstruction and understanding (Chapter 2), and the next part explains the algorithm we developed for fusion lidar and GPS sensors. Chapter 2 illustrates the environment reconstruction technique and the convolution neural network we developed for pedestrian detection in disparity maps. In Chapter 3, we go into details of our fusion method. Result section is included in Chapter 4 . Finally, we conclude and mention some future works in Chapter 5 .

Chapter 2

Environment Reconstruction and understanding

This section explains in detail the algorithms we are using to create and understand the vehicle environment.

2.1 Constructing the vehicle environment from multiple point clouds

In our scenario a Lidar sensor[usually Velodyne] is placed on top of a vehicle that is moving forward, and is continuously rotating and emitting data at 10 fps. Each frame will add new information to the previous frames as well as loose some information existing in the previous ones therefore combining multiple ones will give us a better representation of the scene.

Apparently the above environment reconstruction problem is analogous to the well-known image mosaicking problem in the literature. Similar to image mosaicking, we are also facing the interference from moving objects (in the foreground layer) when attempting to fuse multiple point clouds. Unlike image mosaicking, it is often difficult to find salient feature points (e.g., corners or SIFT keypoints) to facilitate the alignment/registration of 3D point clouds. The only exception seems to be [17] which is based Point Feature Histograms (PFH) but at the price of prohibitive complexity.

Built upon existing point cloud registration tool such as iterated closest point (ICP), we have developed an expectation-maximization (EM)-like technique that can automatically mosaic multiple point cloud sets into a larger one characterizing the still environment surrounding the vehicle and as a byproduct we can get the moving objects in the environment.

We call our algorithm HIERARCHICAL EM-ICP and it consists of two parts the alignment and outlier rejection.

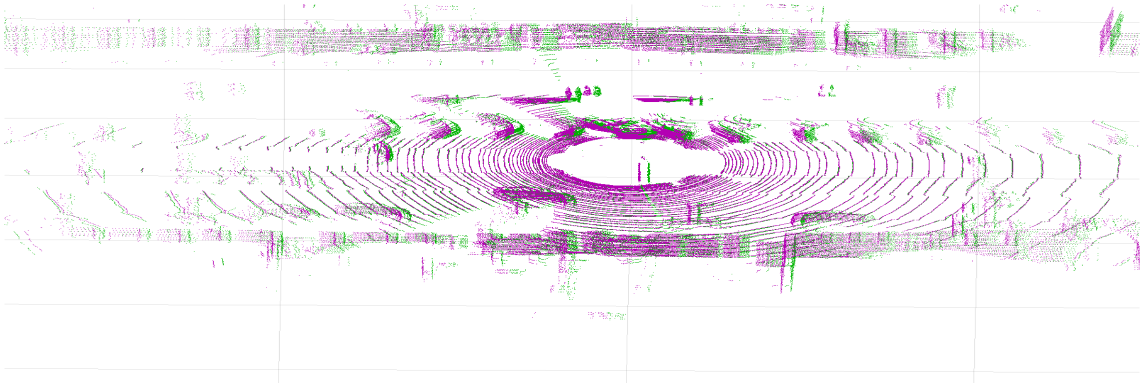


Figure 2.1: Two successive unaligned point clouds, Pk and Pk+1

2.1.1 Simultaneous Alignment of Multiple Point Clouds:Hierarchical Extension of ICP

A widely used tool to align two point clouds is the iterative closest point (ICP) [14] and its variants [1][4][5]. To align multiple (N>2) point clouds P1, P2, . . . , PN, one has to come up with clever strategies of extension. One ad-hoc approach is to sequentially group P1 and P2 into P12 then group P12 and P3 into P123 and so on. In other words, one can think of using P1 as the reference point cloud and align the remaining N-1 point clouds with respect to P1. A moment of thought will conclude this strategy is suboptimal – e.g., due to the motion of self-driving vehicle, P1 and PN could have little overlap and therefore P1 is not an appropriate reference while aligning PN.

Under the assumption that the vehicle is moving at a constant speed, one can derive the optimal reference frame for aligning both P1 and PN should be the frame in the middle. Applying this argument recursively to the first half and second half of P1, P2, . . . , PN, we can obtain the following hierarchical extension of ICP algorithm:

- Perform pairwise alignment to P1, P2, . . . , PN and obtain N/2 point clouds after the merging;

- Recursively apply pairwise alignment to the merged point clouds until only one is left.

The above hierarchical strategy of extending ICP nicely fits octree-based 3D data representation such as Octomap [26]. Tree-based representation offers great flexibility of achieving the tradeoff between complexity and accuracy. For example, one might opt to take the averaging of two aligned point clouds as a coarse-resolution representation (by contrast, the union of them would form a fine-resolution representation). Such hierarchical reconstruction of vehicle environment is beneficial to support various analytical tasks later (e.g., query and matching). To gain a deeper understanding of hierarchical ICP, we propose to analyze its behavior asymptotically – i.e., as the speed of autonomous vehicle decreases to zero, the overlap among adjacent sampling intervals (i.e., Pk vs. Pk+1) would increase implying more computations (larger N) are required for environment reconstruction. To cut back the waste of computation, one can easily skip the pairwise alignment at the first few iterations (equivalent to temporally downsampling the point cloud data set). As the vehicle moves faster, less overlap is introduced which also increases the probability of misalignment.

2.1.2 Outlier Rejection by Robust EM-ICP

The basic idea behind ICP-based alignment [14] is to find out the optimal transformation (e.g., translation and rotation) such that the matching errors between two point clouds can be minimized. To cope with noisy and incomplete observation data, robust ICP based on Least Trimmed Square was developed in [27]. Under the context of intelligent vehicles, outliers in point cloud arise from two primary sources: one is the moving objects (e.g., walking pedestrians, moving vehicles and motorcycles); the other is the objects located at the border of the scene (i.e., analogous to covered/uncovered pixels in image mosaicing).

Inspired by previous work on robust surface registration [5], we propose to robustly align two point clouds with outlier rejection. The key motivation is based on the observation that point cloud registration and outlier detection have characteristics of chicken-and-egg problem – i.e., solving one immediately unlocks the solution to the other. Formally, we might treat unknown transformation parameters and class labels (whether a point is inlier or outlier) as a pair of peer hidden variables. Such perspective allows us to leverage classical expectation-maximization (EM) method into ICP as follows (similar treatment can be found in [28]).

- E-step: For an estimated transformation, align two point clouds and calculate the closest distance for each point. Declare a point to be an outlier if the closest distance is above a pre-selected threshold
- M-step: For a given set of inliers, apply ICP to update the estimate of transformation parameters (i.e., translation and rotation).

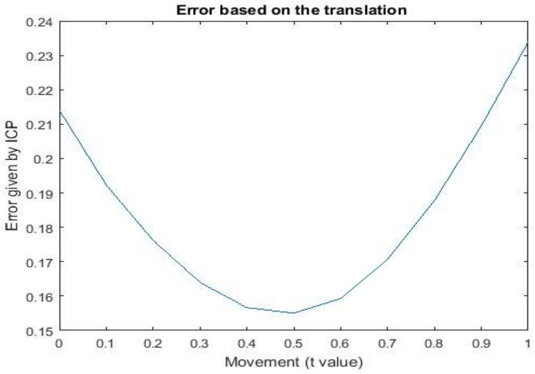


Figure 2.2: The distribution of matching errors in ICP algorithm as a function of displacement (outliers are declared by those a certain threshold for example .12m).

2.1.3 Segmentation of the background

For a better representation of the vehicle environment, We propose to cluster our environment using algorithm such as DBSCAN [29]. The result is presented in the following figure where objects are color coded.

2.2 Constructing depth maps from the point cloud data

In the previous section we were interested in constructing the environment and dividing it into inliers and outliers. In this section we will try to better understand the environment by trying to detect and recognize the objects that constitute said environment. Such knowledge is crucial for many applications of autonomous vehicles.

We will generate depth maps from the point cloud and apply a convolution neural network on them.

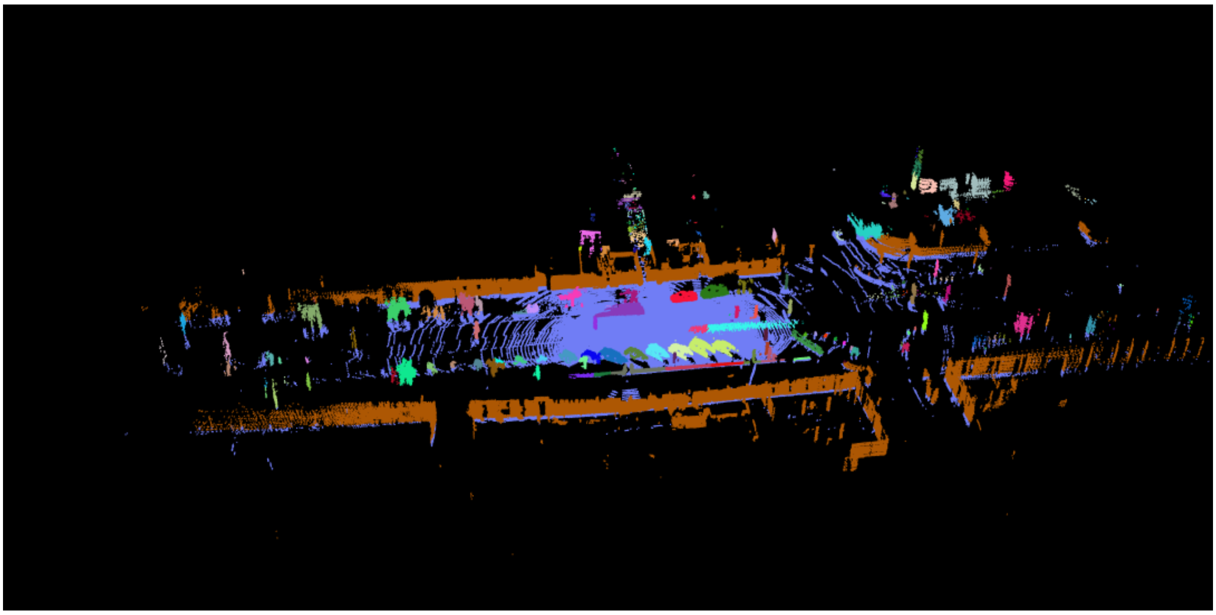


Figure 2.3: Segmentation of still background objects in a complex scene (different objects are highlighted by different colors).

2.2.1 Depth maps creation

Our method for depth map creation is based on the following work [30].

First we eliminate the points in the data with a depth value (x direction in world coordinate) inferior to 0 meter; then we transform the rest of the points from world coordinate to camera coordinate using transformation matrix and then we project to image plane using camera calibration matrices; both matrices are already given. After the projection each point in the point cloud will have pixel coordinates; these coordinates can be floating values.

Let's suppose a point after projection has the px and py pixel coordinates. To create our Sparse map I, we will give a pixel which coordinates are between [px-k,px+k] and [py-k,py+k] (where k is a fixed kernel) the normalized value of the x component of the point (depth; normalized between [0,255]); we apply this to each point in the data. After that we apply bilateral filter (equation 3.1) to smooth the image and get the final depth map D.

$$\frac{1}{W_p}\sum_{q\in N} G_{\sigma_s}((||p-q||)G_{\sigma_r}(|I_q|)I_q \tag{2.1}$$

Where G_{σ_s} weights points q inversely to their distance to position p,G_{σ_r} penalizes the influence of points as function of their range values, and finally W_p is a normalization factor that ensures weights sum to one,i.e., $W_p = \sum_{q\in N} G_{\sigma_s}((||p-q||)G_{\sigma_r}(|I_q|)$.

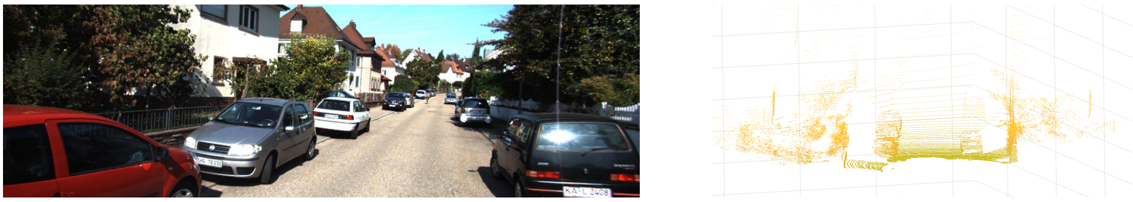


Figure 2.4: RGB and point cloud data

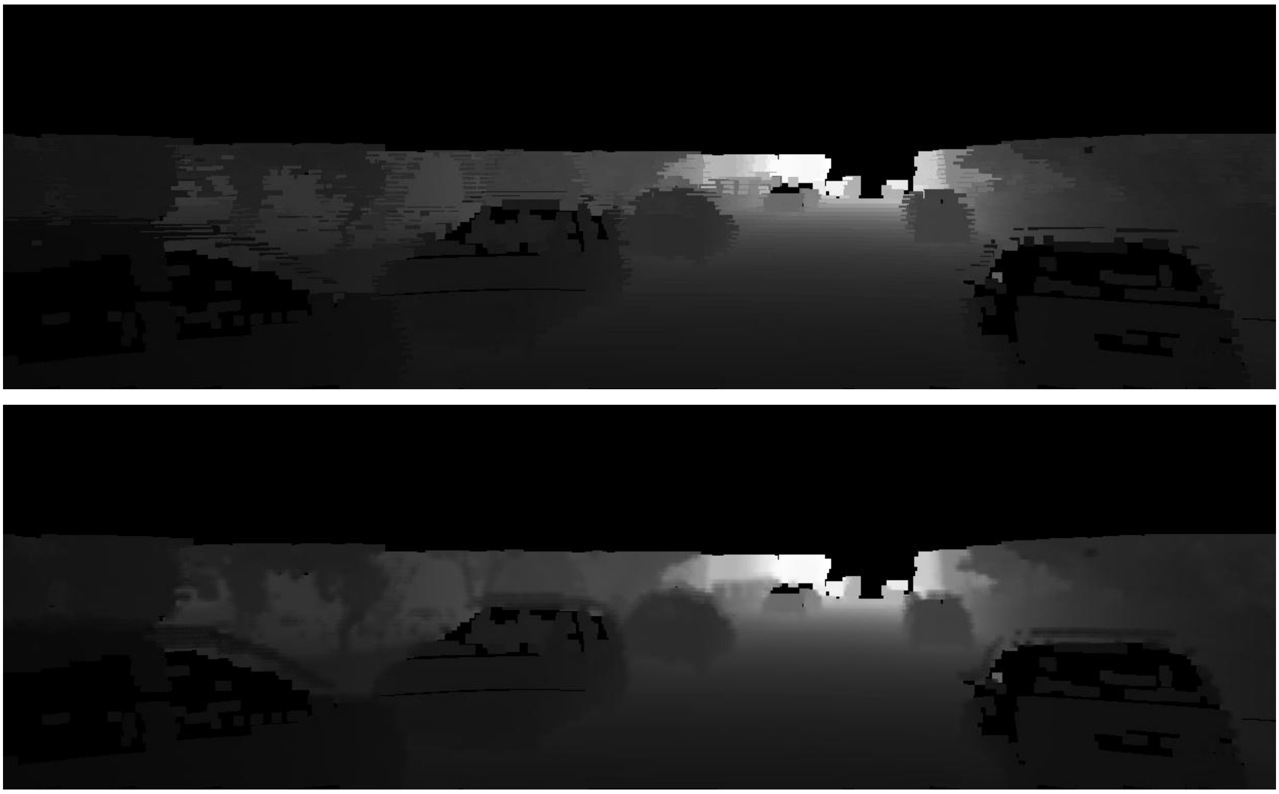


Figure 2.5: Sparse I map and final depth map D

We can imagine that instead of using the x component of each point to create the depth map, we can use the y and z components; if we do so on the same point cloud we get the two following depth maps respectively.

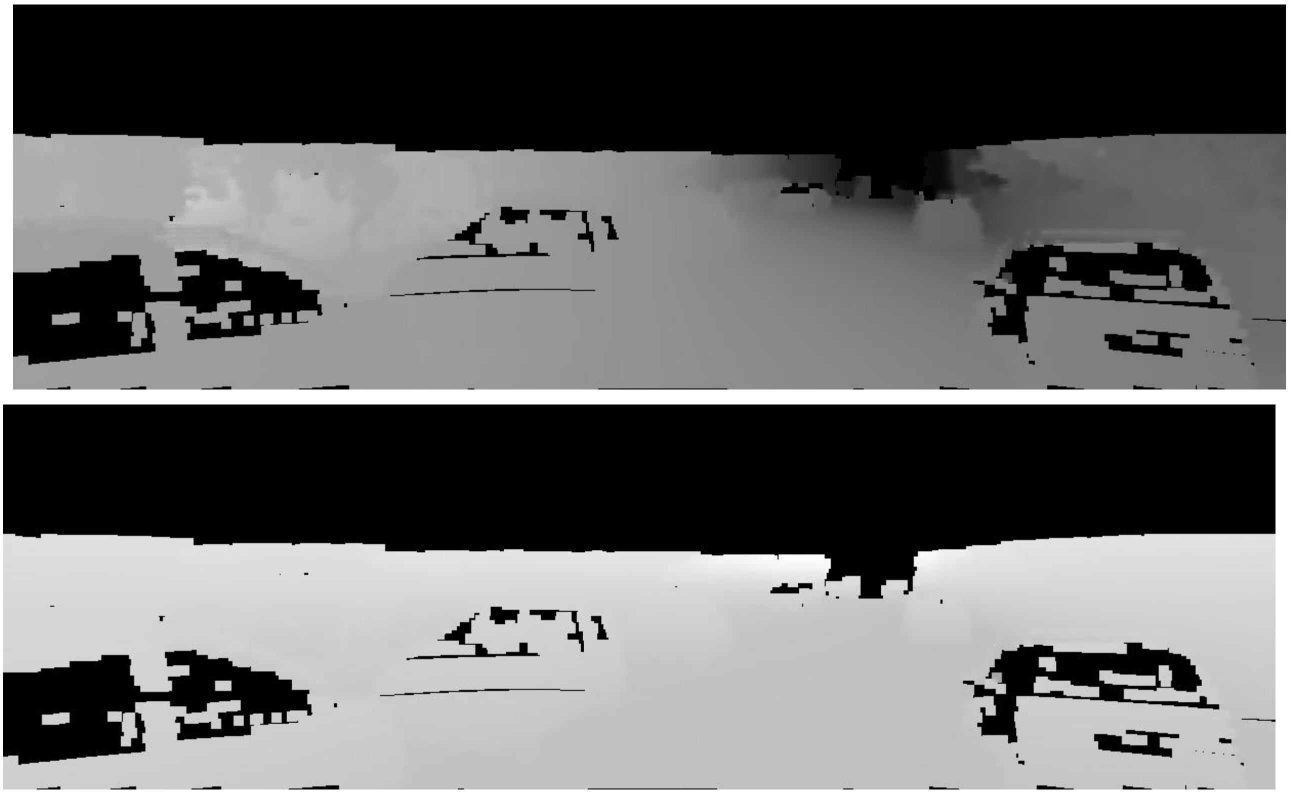


Figure 2.6: Depth generated respectively with the y and z components

As a byproduct of our depth map creation and due to the perfect alignment between depth map and the RGB image; we are able to assign to each point cloud an RGB value and create a 6 dimension XYZRGB data. (figure 3.6 illustrates that).

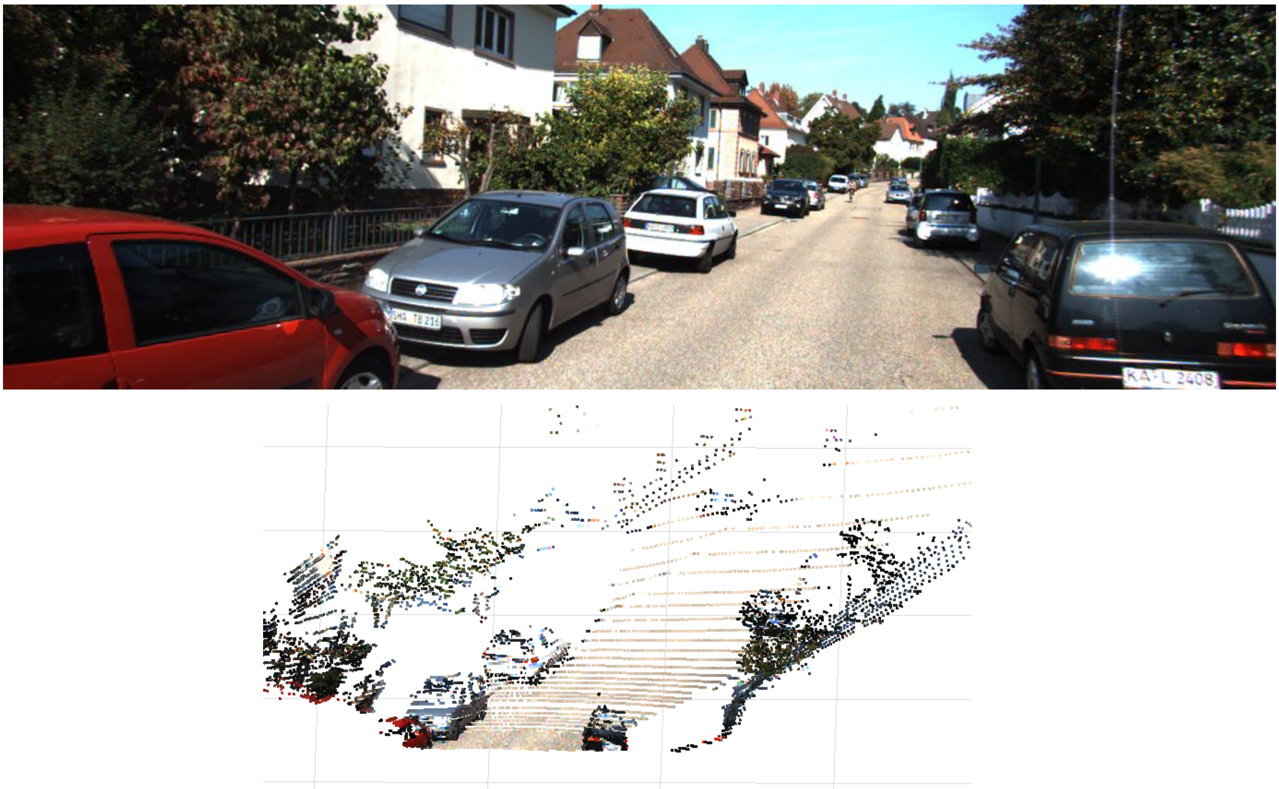


Figure 2.7: the RGB data and the XYZRGB data for comparison

Furthermore we can use our alignment method on the multiple XYZRGB data to create a vehicle environment that contains the color information for each figure. When we do that we have the following results in figure 3.7.

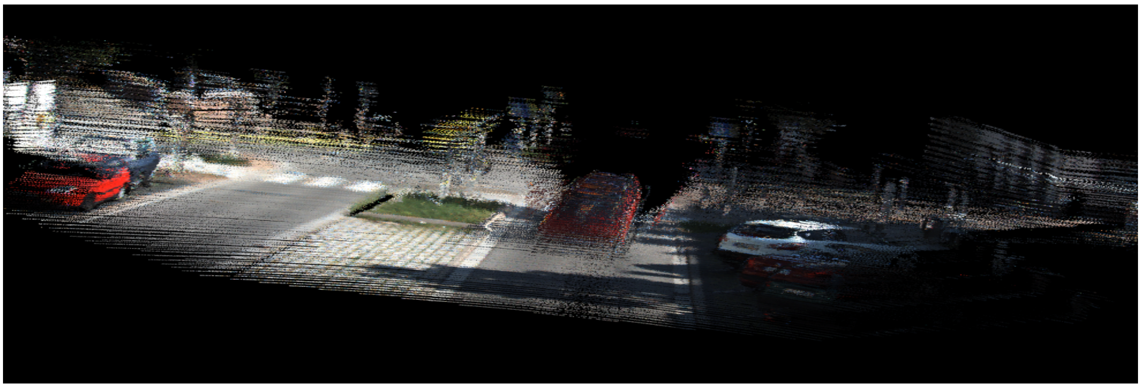


Figure 2.8: Results after combining multiple XYZRGB data to create the vehicle environment

In this section we showed how to create the depth map now, we will move on to the task of object detection specifically pedestrian detection.

2.2.2 Using convolution neural network on depth map for the purpose of pedestrian detection

Convolution neural network is the state of art machine learning technique for object recognition and detection in images. A Convolutional Neural Network (CNN) is comprised of one or more convolutional layers (often with a subsampling step) and then followed by one or more fully connected layers as in a standard multilayer neural network. The architecture of a CNN is designed to take advantage of the 2D structure of an input image (or other 2D input such as a speech signal). This is achieved with local connections and tied weights followed by some form of pooling which results in translation invariant features. Another benefit of CNNs is that they are easier to train and have many fewer parameters than fully connected networks with the same number of hidden units. In this article we will discuss the architecture of a CNN and the back propagation algorithm to compute the gradient with respect to the parameters of the model in order to use gradient based optimization.

We will be working with KITTI dataset [31] to generate our training and testing data for the experiment.

a- Generating training and testing data from KITTI dataset

Kitti database contains about 7000 point cloud data files; each file has a corresponding labeled RGB image. We use these labels to extract the ground truths from the depth maps because a depth map is perfectly aligned with the RGB image.

To generate the training and testing data for the CNN we use the sliding window approach and process each window to see it is a positive or negative sample and resize it to 64x64.

The processing is done in the following way:

- Generating positive samples: If the overlap between the window and the bounding box given by the labeling is greater or equal to 0.5; consider as positive sample and also include the original bounding box.
- For negative samples we have three categories:
  1.Hard negative: Take four samples randomly with overlapping <0.4 and >=0.3.
  2. Medium negative : Take three samples randomly with overlapping < 0.3 and > 0.
  3. Easy negative : Take 2 samples randomly with overlapping equal 0.

Therefore we have 9 negative samples for 2 positive sample and in total we have around 4000 positive samples and 2000x9 negative samples

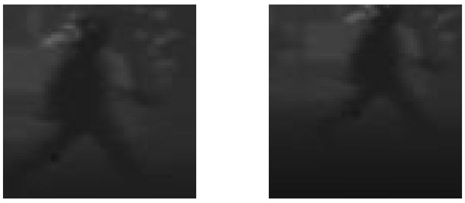


Figure 2.9: Positive samples

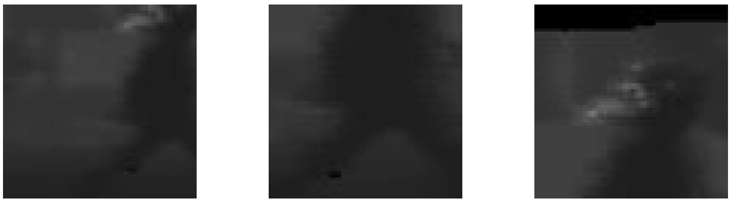


Figure 2.10: Hard negative samples

We take 80 for training and 20 for testing, The ratio of positive to negative is 2:9 in the training phase; And 1:1 in the testing phase.

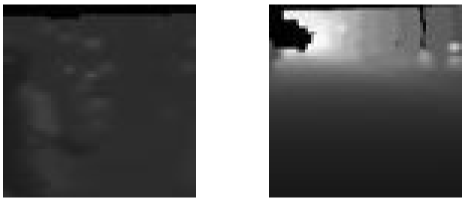


Figure 2.11: Medium and Easy negative samples respectively

Now we are going to move to the discuss the detail of our CNN.

b-CNN architecture

The CNN architecture is as follow using the following notation.

Input Grayscale image of size 64x64.

k = kernel size; f = number of feature maps; p = padding value and s = stride.

C(k,s,p,f) represents a convolution layer.

P(k,s,p) represents pooling layer.

F(N) represents a fully connected layer with N output

- C(k=5,s=1,p=0,f=16) and Relu; output of size 60x60
- P(k=2,s=2,p=0) ; output of size 30x30
- C(k=5,s=1,p=0,f=16) and Relu; output of size 26x26
- P(k=2,s=2,p=0); output of size 13x13
- C(k=3,s=3,p=0,f=16) and Relu; output of size 6x6
- P(k=2,s=2,p=0); output of size 3x3
- C(k=3,s=1,p=0,f=16) and Relu; output is 16 scalar values
- F(2); output 2 scalar values
- Softmax Classifier; output 2 scalar representing the probability of each class.

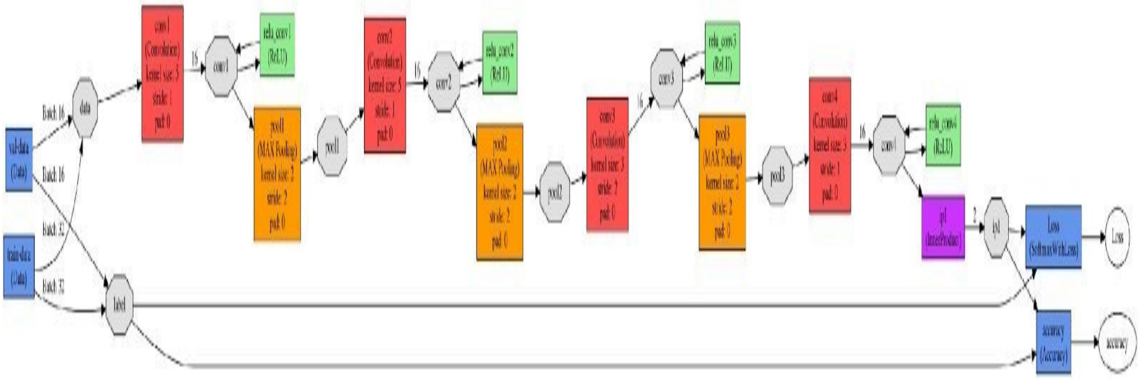


Figure 2.12: Graphical representation of CNN using a Caffe[1] tool

c-CNN hyperparameters

For training and testing we use caffe [32] with the following hyper-parameters for our network:

Test iteration	100
Test interval	200
Learning rate	1e-03 for variable stride and then 1e-05 for fixed stride of 32.
maximum ieration	200000
learning policy	polynomial
Power	3.0
momentum	0.9
weight decay	1e-3
machine	CPU

Chapter 3

Fusion of Information from Lidar sensor and V2X Communicated Data for Automated Driving

Sensor fusion is the process of combining data from different sensors to get a result that is better that any one sensor (lower error variance for example). There exists many proposed more formal definition,in [33] Wald uses “data fusion” for a formal framework that comprises means and tools for the alliance of data originating from different sources that aims at obtaining resulting information of superior quality; however the exact definition of superior quality depends on the application. The term “data fusion” is used in this meaning by the Geoscience and Remote Sensing Society, by the U. S. Department of Defense [69], and in many papers regarding motion tracking, remote sensing, and mobile robots. Unfortunately, the term was not used in the same meaning in the last years, in some models data fusion denotes the fusion of raw data[18].

To avoid the confusion Dasarathy[[34] distinguish the term information fusion from sensor fusion. Information fusion encompasses theory, techniques and tools conceived and employed for exploiting the synergy in the information acquired from multiple sources (sensor, databases, information gathered by human, etc.) such that the resulting decision or action is in some sense better (qualitatively or quantitatively, in terms of accuracy, robustness, etc.) than would be possible if any of these sources were used individually without such synergy exploitation. While sensor fusion, is the combination of sensory data and data derived from sensors.

One sensor exhibits many limitations : sensor deprivation due to the breakdown of a sensor; limited spacial coverage usually an individual sensor covers only a restricted region, limited temporal coverage; imprecision and uncertainty. In turn, using sensor fusion we can expect many gains like [35] [36]: robustness and reliability, extended spatial and temporal, increased confidence, robustness against inference...etc.

However sensor fusion is not without limitations; Fowler stated a harsh criticism in 1979: One of the grabbiest concepts around is synergism. Conceptual application of synergism is spread throughout military systems but is most prevalent in the "multisensor" concept. This is a great idea provided the input data are a good quality. Massaging a lot of crummy data doesn't produce good data; it just requires a lot of extra equipment and may even reduce the quality of the output by introducing time delays and/or unwarranted confidence. [. . .] It takes more than correlation and fusion to turn sows' ears into silk purses. [31, page 5].

Many have tried to prove the opposite, Nahin and Pokoski [37] presented a theoretical proof that the addition of sensors improves the performance in the specific cases for majority vote and maximum likelihood theory in decision fusion. Performance was defined as probability of taking the right decision without regarding the effort on processing power and communication.

We distinguish the following methods of sensor fusion: Smoothing, Filtering, and Prediction which refers to the task of fusion different noisy measurements construct the parameter of interest.The stochastic Kalman Filter [38] uses a mathematical model for filtering signals using measurements with a respectable amount of statistical and systematical errors. Generally; The filter uses a discrete-time algorithm to remove noise from sensor signals in order to produce fused data that, for example, estimate the smoothed values of position, velocity, and acceleration at a series of points in a trajectory. Inference methods specifically bayesian inference [6] based on bayes theorem quatify the probability of hypothese H given that an event E has occured. Examples for applications based on Bayesian inference can be found in [39] for merging multiple sensor readings, in automatic classification of sensor inputs (e. g., the computer program AUTOCLASS [40] developed by the NASA), or in map building for mobile robots [41]. However when bayesian inference is used for sensor fusion certain drawbacks can happen [42].

3.1 Proposed Framework

Our method fuses data from local sensor and data obtained from outside. Automated Vehicles use a number of sensing technologies, such as Lidar, Radar, and Cameras, which all rely on "line of sight" (LOS) view and favourable environmental conditions. Lack of these conditions leads to safety issues. Wireless communication is essential to acquire non line-of-sight view and information regarding objects beyond the area directly sensed by local sensors . Such communicated data may even be more precise than what local sensors can observe in line of sight. Data Fusion mechanisms should be devised to allow fusion of information from local and remote (V2X communicated) sensors.

Each car combines information (fig 3.1) received from different sources to create and update a local map of objects around it:

Information derived from local sensing modalities Previous map knowledge about the driving environment information derived through processing data sent by other cars.

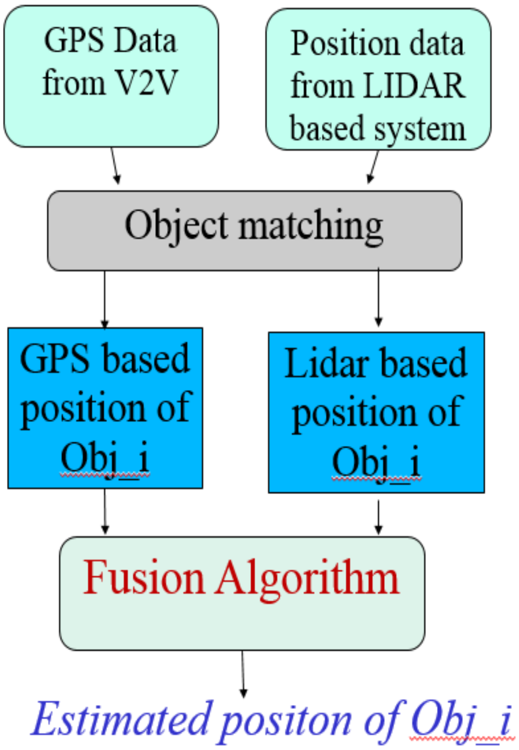

GPS Data from V2V
Position data from LIDAR based system
Object matching
GPS based position of Obj_i
Lidar based position of Obj_i
Fusion Algorithm
Estimated positon of Obj_i

Figure 3.1: Fusion Framework

3.2 fusion algorithm

The fusion algorithm generates a position estimate using a fusion of information from different sources. For example, a linear combination of detected distance. Fusion can happen at different levels: Processing of raw data from different sources to determine the position or Fusion of point-wise positions from different sensors (this work).

Figure.3.2 represents a rough sketch of how the error of the two different sensors compares.

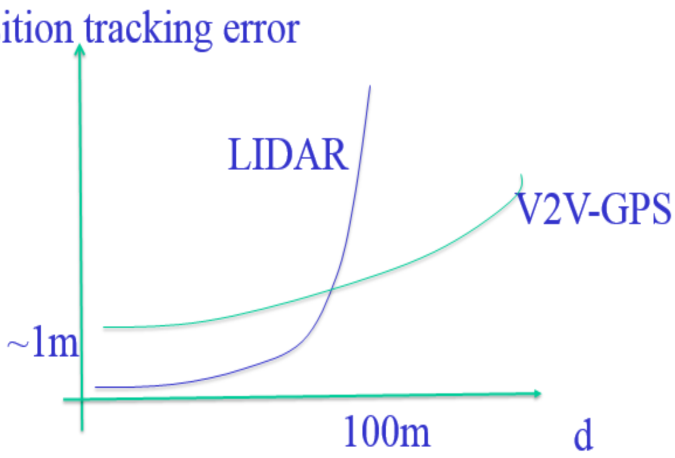


Figure 3.2: A rough sketch of how the different methods compare

3.3 Fusion method

We use a linear fusing method.Assume the position (distance) from LIDAR and GPS over Dedicated Short Range Communications(DSRC) are denoted as X_L, and X_G. Assuming error can be represented in a Gaussian form, for actual distance X we have:

$$X_G = X + E_G(X) \quad \text{and} \quad X_L = X + E_L(X) \tag{3.1}$$

$E_G(X)$ and $E_L(X)$ are the error terms with variance $\sigma_G(X)$ and $\sigma_L(X)$ which increase with distance X.

A linear fusion method as follows is proven to provide lower error variance if a is selected properly (assuming the errors are normally distributed):

$$\widehat{X} = aX_G \quad + \quad (1-a)X_L \tag{3.2}$$

The error variance of the fused result is defined as follow:

$$\sigma^2(\widehat{X}) = a.\sigma_G^2(X) \quad + \quad (1-a).\sigma_L^2(X) \tag{3.3}$$

To find appropriate weights, we need to calculate error profiles of detection methods

3.4 Detection Error Profile

We assume that the error of the lidar based method will increase with the distance of the object from the sensor and V2X error should remain relatively constant.

3.4.1 Lidar method error profile

For LIDAR based system we take the example of CSoR (CSHOT on Laser Reflectance) by Leonard Plotkin [43]. Leonard Potkin propose a vehicle detection method using only Lidar data. This method uses data from KITTI [31]. We use the KITTI [31] groundtruth for to get the error profile.

Figure 3.3 shows the result of his detection method when applied to the corresponding RGB images and shows the corresponding point cloud data frame.

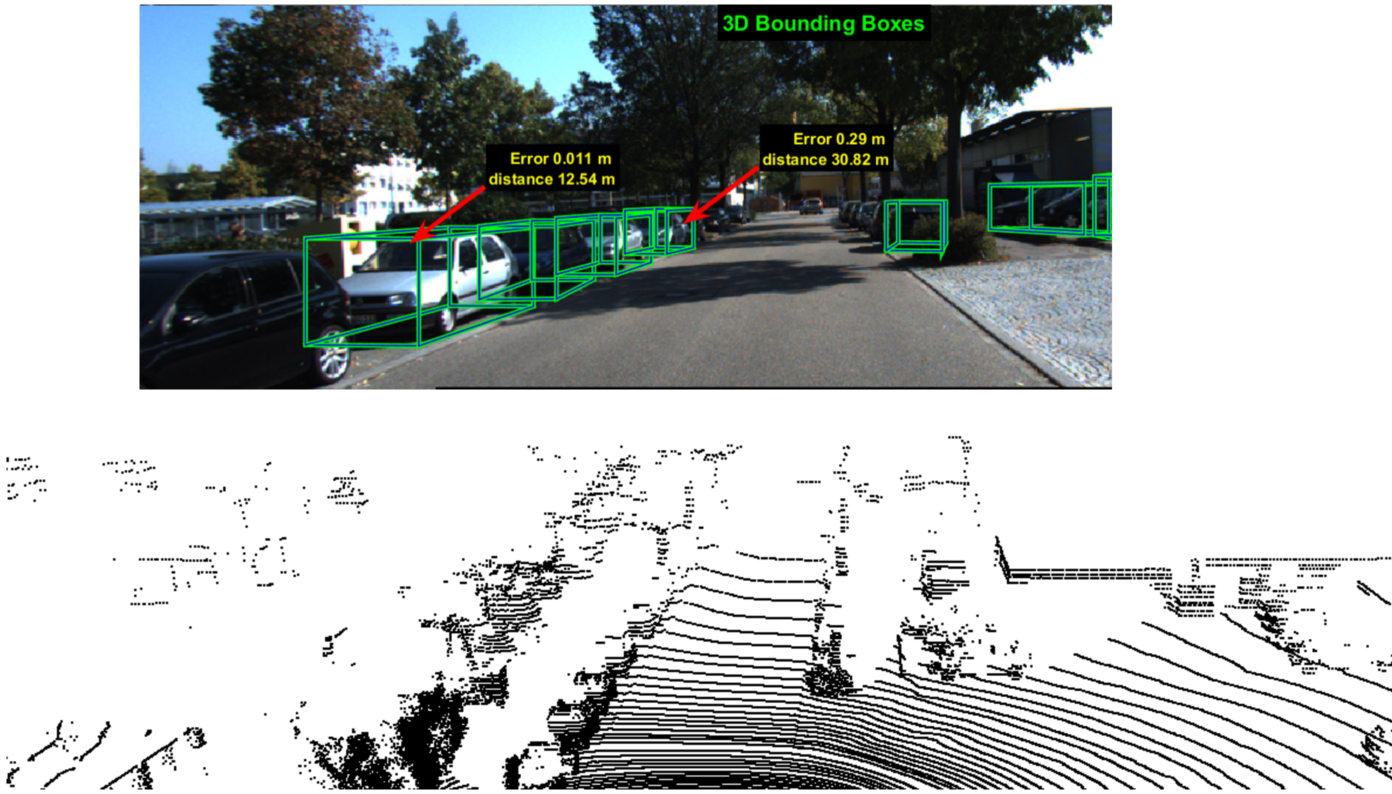


Figure 3.3: Examples of the point cloud data from LIDAR, as well as a snapshot of a scene and detected vehicles are shown at the top

Leonard Plotkin [43] uses k-means, Adaboost [?] and KNN with CSHOT [44] features to estimate the object position. When we run the algorithm we find that it gives some negative detection samples; to choose the right detection sample we consider the values with a bounding box that has intersection with the groundtruth ¿=0.5.

Figure 3.4 shows the scatter plot between the real X values and the predicted ones. X is the x-axis in camera coordinates.

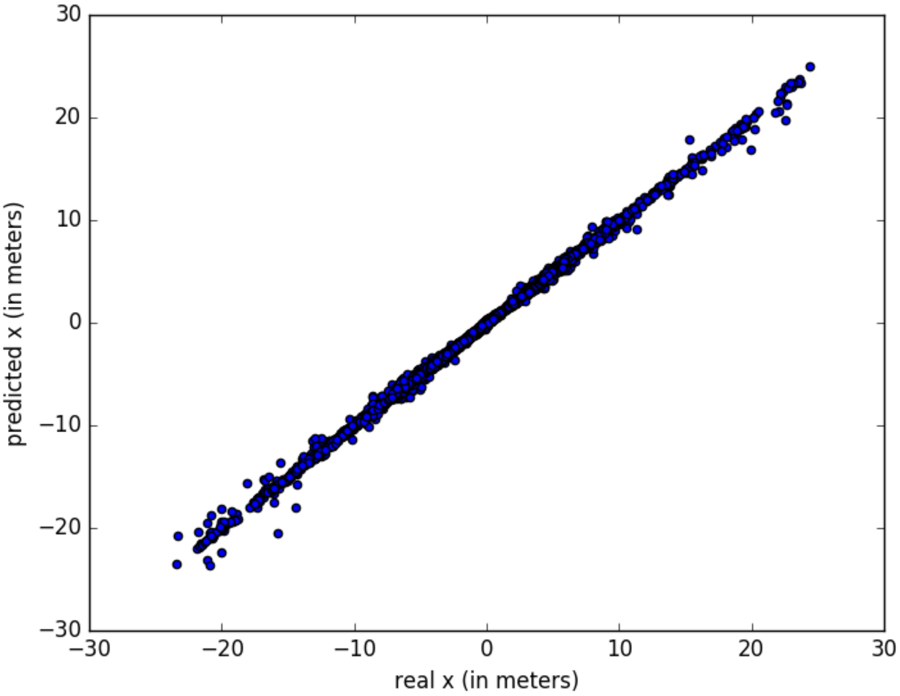


Figure 3.4: Scatter plot of X predicted and X real

Figure 3.5 shows the scatter plot between the real Z values and the predicted ones. Z is the z-axis on the camera coordinate

We also plot the histogram based on the euclidean error value for all the data (Fig.3.6).

We can see from the histogram that the vast majority of the samples exit where the error is between 0 and 1 meters. Therefore it might be a good idea to try to plot the histograms where the error is between 0 and 1 m and errors superior to 1.

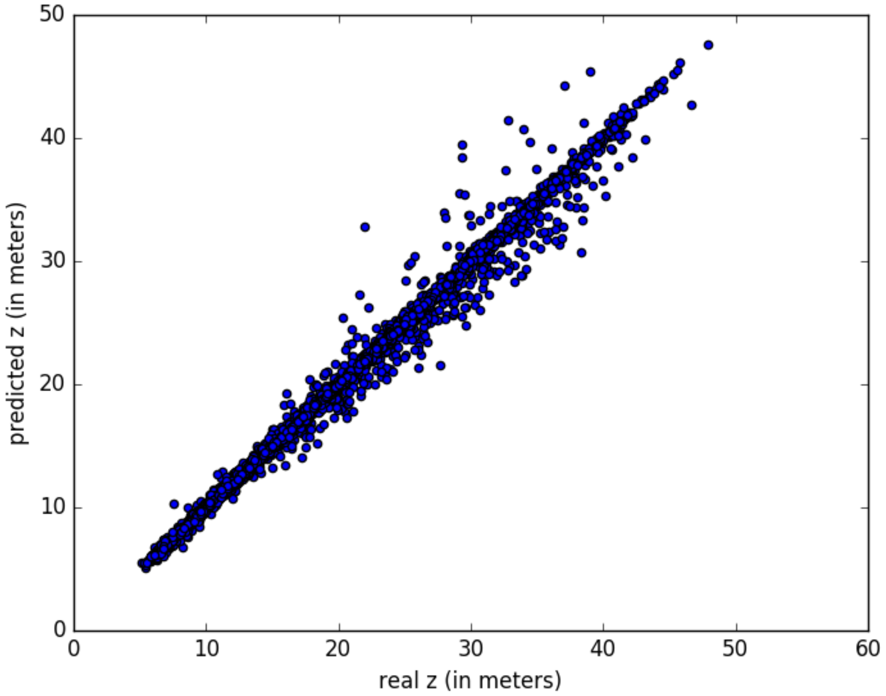


Figure 3.5: Scatter plot of Z predicted and Z real

Fig. 3.7 shows the histogram for errors values between 0 and 1. and Fig 3.8 shows the corresponding PDF.

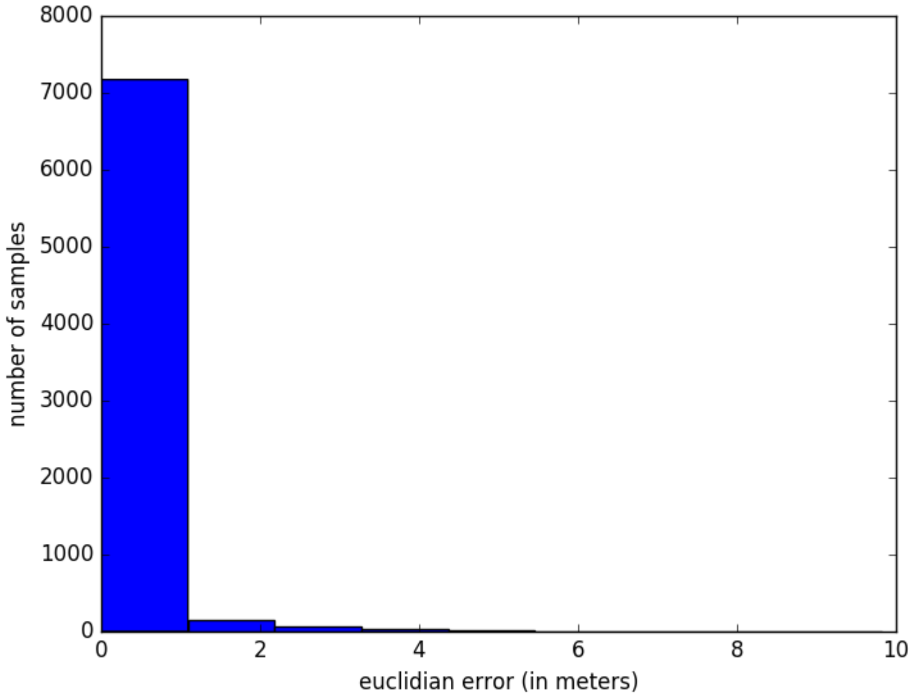


Figure 3.6: Histogram of the error

Fig 3.9 shows the histogram for the errors superior to 1m.

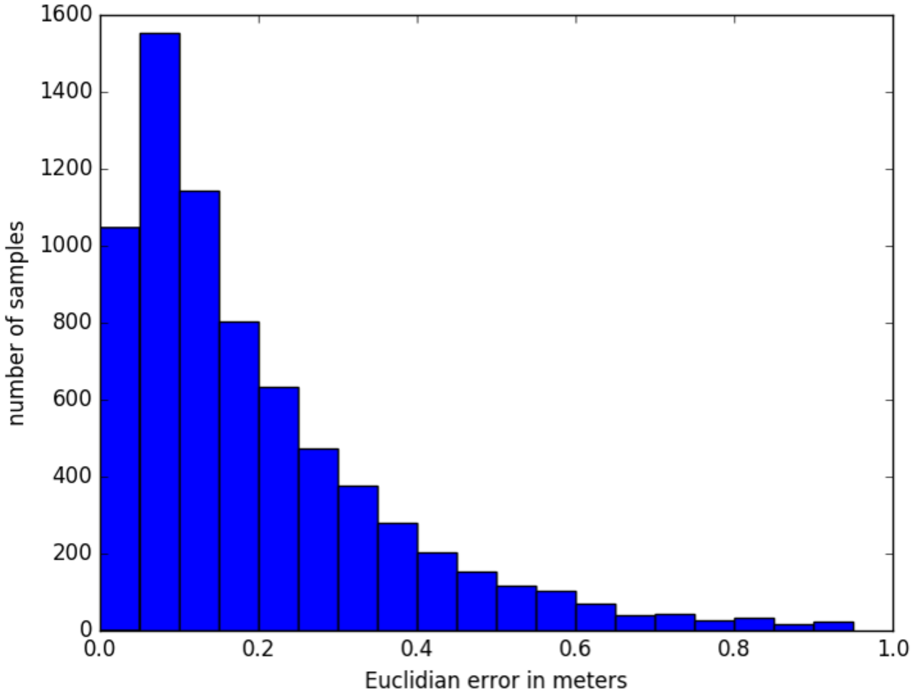


Figure 3.7: Histogram of the error values between 0 and 1

For a better understanding of the error profile we also analyze the 95 percentile. Fig.3.10 is a table containing the values of each percentile with the range representing the distance from the detected vehicle and the number of samples in each range, for all errors.

Fig.3.11 shows plot the 95 percentile for all errors

Fig 3.12 shows a table containing the values of each percentile with the range representing the distance from the detected vehicle and the number of samples in each range, for errors between 0 and 1 m.

Fig.3.13 shows plot the 95 percentile for errors between 0 and 1m.

Fig 3.14 shows the trend of the error.

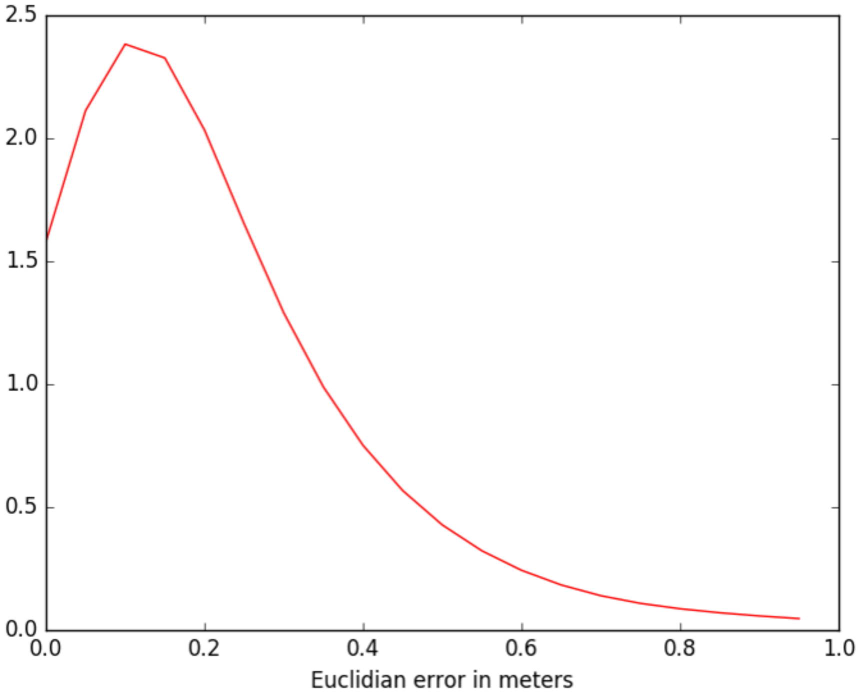


Figure 3.8: PDF of the error values between 0 and 1

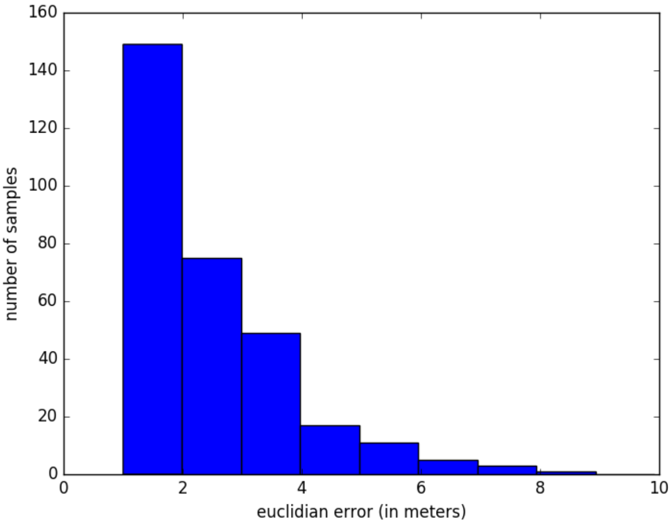
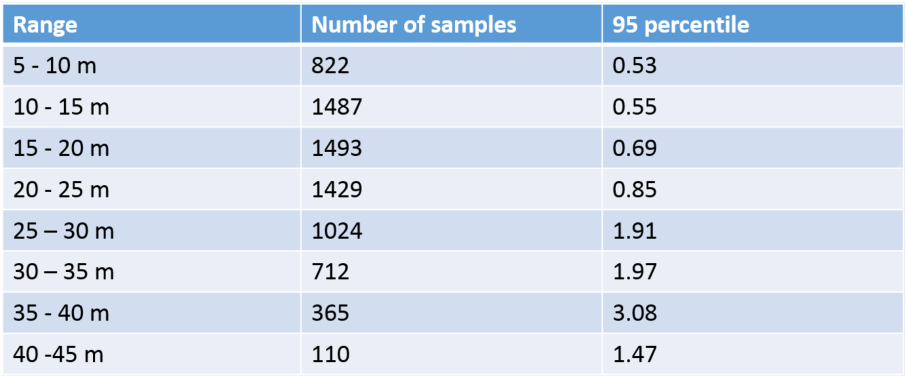


Figure 3.9: Histogram of error values superior to 1m



Range	Number of samples	95 percentile
5 - 10 m	822	0.53
10 - 15 m	1487	0.55
15 - 20 m	1493	0.69
20 - 25 m	1429	0.85
25 – 30 m	1024	1.91
30 – 35 m	712	1.97
35 - 40 m	365	3.08
40 -45 m	110	1.47

Figure 3.10: table containing the 95 percentile for all error values

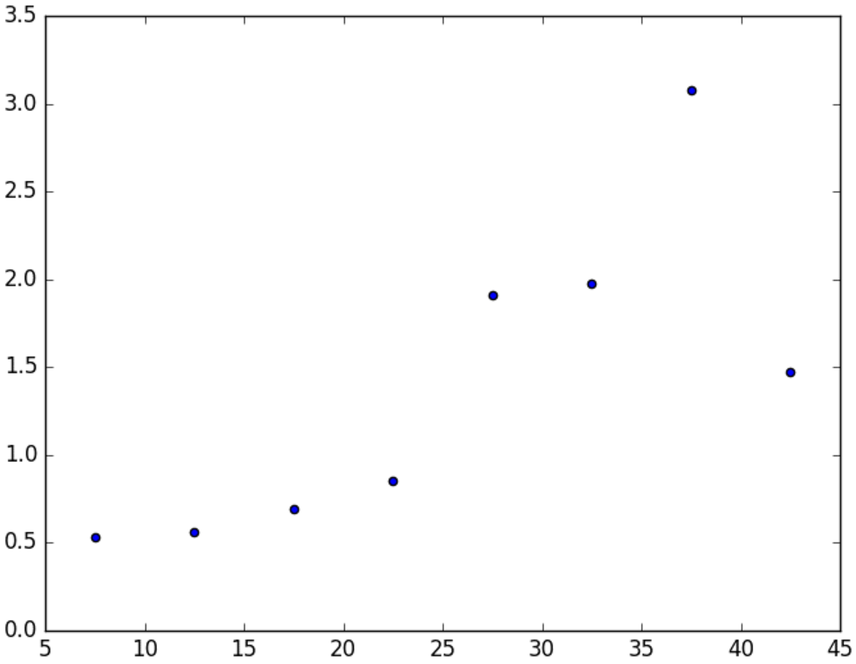
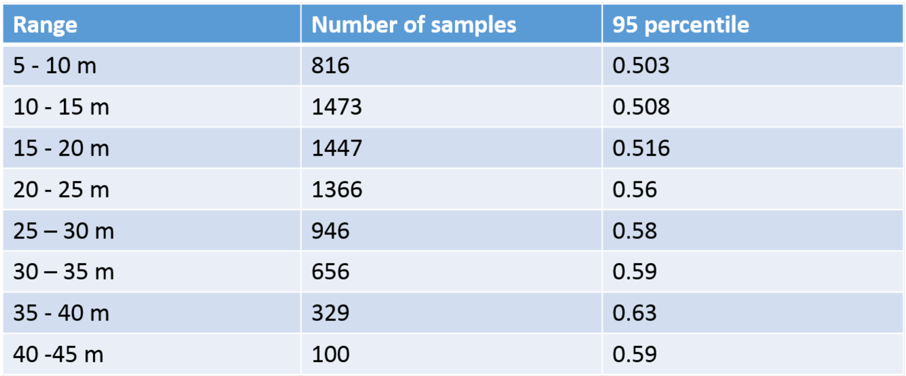


Figure 3.11: plot of the 95 percentile for all error values



Range	Number of samples	95 percentile
5 - 10 m	816	0.503
10 - 15 m	1473	0.508
15 - 20 m	1447	0.516
20 - 25 m	1366	0.56
25 – 30 m	946	0.58
30 – 35 m	656	0.59
35 - 40 m	329	0.63
40 -45 m	100	0.59

Figure 3.12: table containing the 95 percentile for error values between 0 and 1m

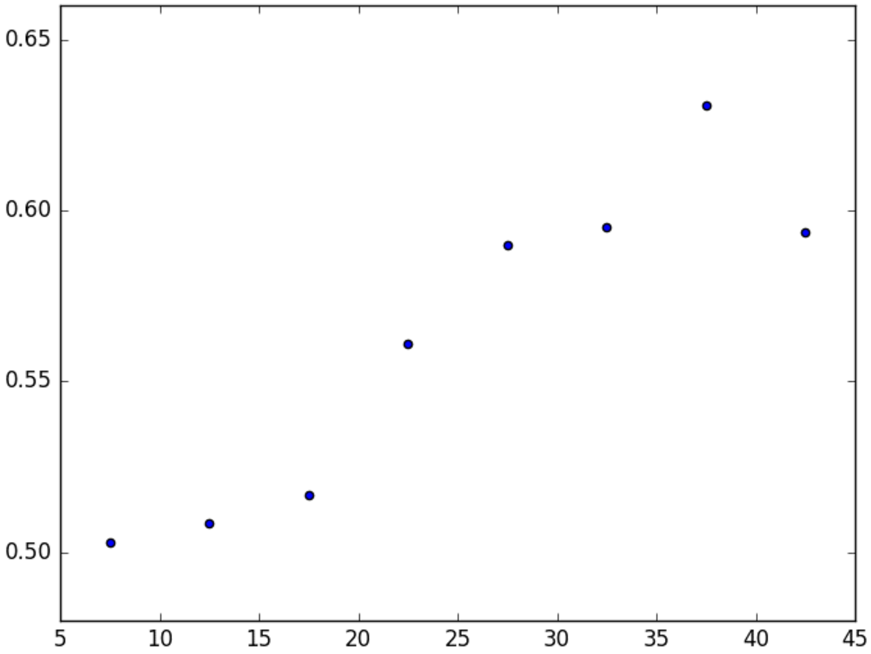


Figure 3.13: plot of the 95 percentile for error values between 0 and 1m

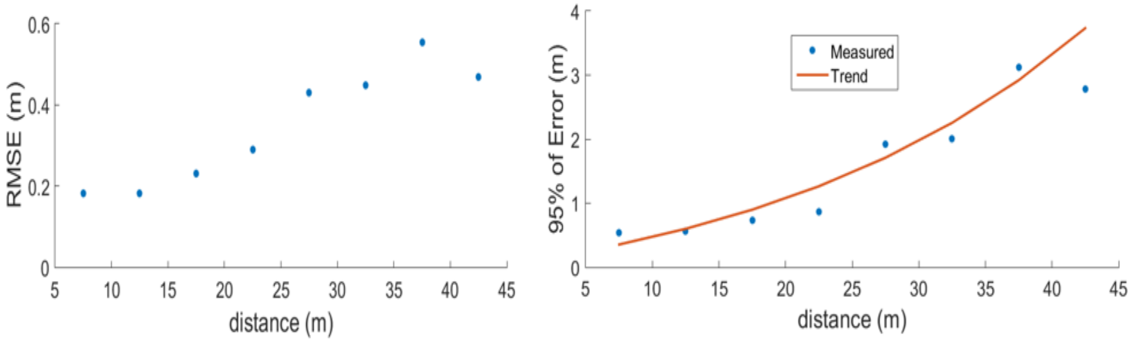


Figure 3.14: Error profile of Lidar based method

3.4.2 GPS error profile

The Global Positioning System (GPS),is a space-based radionavigation system owned by the United States government and operated by the United States Air Force. It is a global navigation satellite system that provides geolocation and time information to a GPS receiver anywhere on or near the Earth where there is an unobstructed line of sight to four or more GPS satellites. We will focus on differential GPS sensors; Differential Global Positioning System (DGPS) is an enhancement to Global Positioning System that provides improved location accuracy, from the 15-meter nominal GPS accuracy to about 10 cm in case of the best implementations. DGPS uses a network of fixed ground-based reference stations to broadcast the difference between the positions indicated by the GPS satellite systems and the known fixed positions. These stations broadcast the difference between the measured satellite pseudoranges and actual (internally computed) pseudoranges, and receiver stations may correct their pseudoranges by the same amount. The digital correction signal is typically broadcast locally over ground-based transmitters of shorter range.

Information in Basic Safety Messages is mainly derived from GPS sensors. and With Differential GPS sensor, error variance is under 1 m: $\sigma_G(X) \sim 0.6$m for all X.

3.4.3 Defining the weights

For point-wise fusion of remote vehicle position from GPS and LIDAR, we propose distance dependent weight of :

$$a = (\sigma_L(X)/\sigma_G(X)) * C; \quad C < max(\sigma_G(X)/\sigma_L(X)) \tag{3.4}$$

The constant C ensures that a<1.

Chapter 4

Results and Analysis

In this section we will presents the results of our environment reconstruction and understanding plus our sensor fusion method. We tested our approaches on two publicly available lidar and RGB databases.

4.1 EM-ICP algorithm

In this section, we report our experimental results on KITTI data set [31]. The LIDAR sensor used in [31] is Velodyne HDL-64E (64 channels and range of 120m). The height of LIDAR scanner is 1.73m and spins at 10 frames per second (capturing approximately 100k points per cycle).

In the first set of experiments, we report the environment reconstruction results of the proposed EM-ICP for a variety of point clouds – urban vs. rural (refer to Fig. 4.1) and road vs. intersection (refer to Fig. 4.2). In our experiments, we have aligned and combined N=40 point clouds (each cloud contains around 200,000 points). It can be observed that the density of point cloud increases along the moving direction of HV. This is consistent with the fact that more overlaps exist in the point clouds in the front of than behind the vehicle. We also notice that the reconstruction of urban environment appears to be more accurate than that of rural one. One possible explanation is that man-made structures in urban environments (e.g., buildings, traffic signs etc.) better fit the proposed EM-ICP algorithm than those natural objects such as trees in rural environments. Indeed, similar findings can be found in the previous work on efficient variants of ICP algorithms [45].

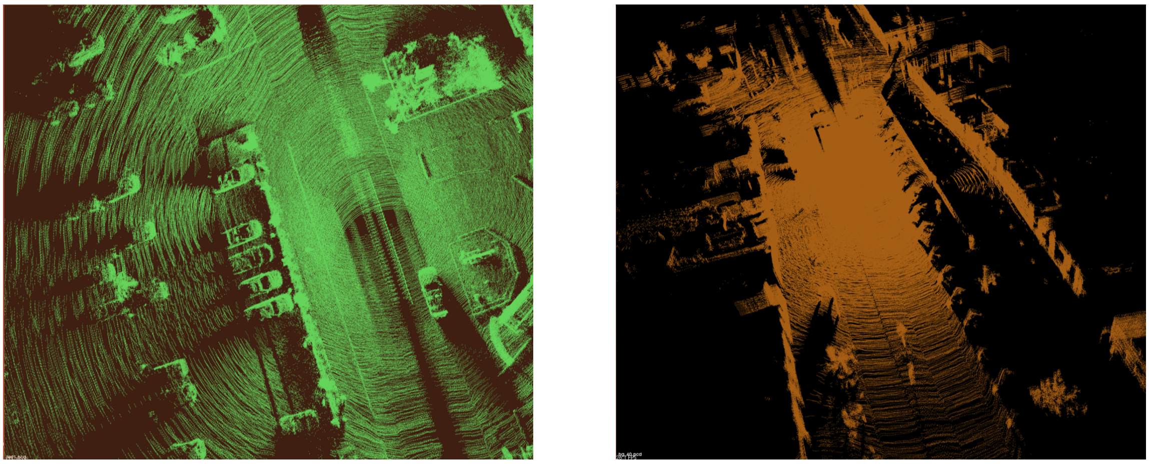


Figure 4.1: Urban environment reconstruction from Ford and KITTI data

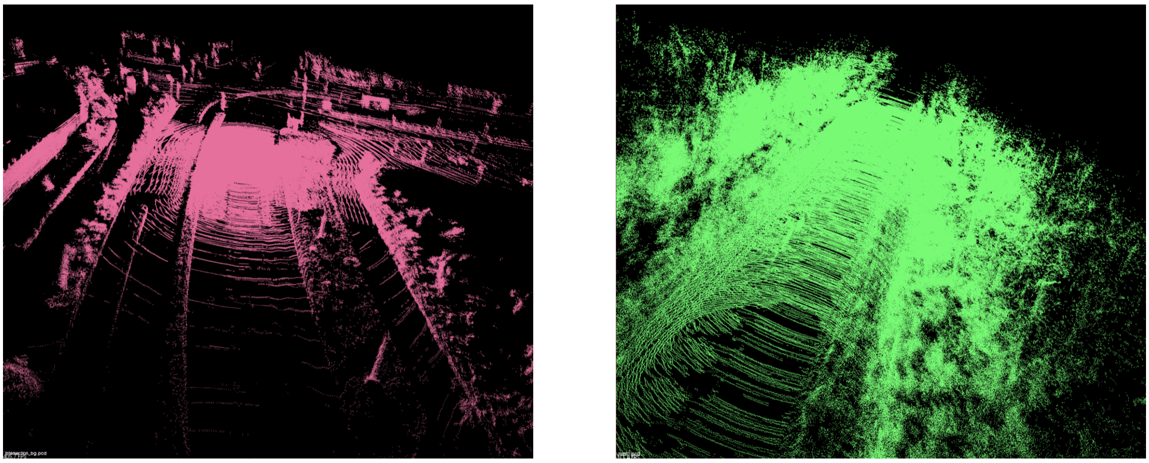


Figure 4.2: Road intersection plus rural environments reconstruction from KITTI data

To qualitatively evaluate the accuracy of our EM-ICP algorithm, we have used the ground-truth (GT) data supplied by KITTI database [31]. In Fig. 4.4), we have plotted the comparison of predicted vehicle speed and GT – note that no filtering has been applied to the measurement speed. Assuming the errors of predicted speed can be modeled by Gaussian, some low-pass filtering can be used to significantly improve the accuracy of speed prediction. In Fig. 4.4), we have plotted the comparison between the predicted and actual x component of translation characterizing how much the sensor has moved from frame to frame in the x direction. It can be observed that the prediction is fairly close to the GT values justifying the accuracy of the proposed EM-ICP alignment method.

In figure 4.3, We show an example of the extracted foreground, where the overlay indicate

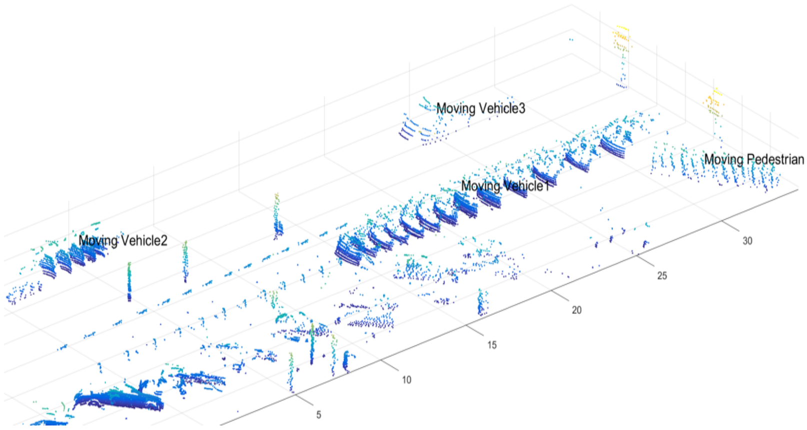


Figure 4.3: Example of Extracted Foreground

the moving objects.

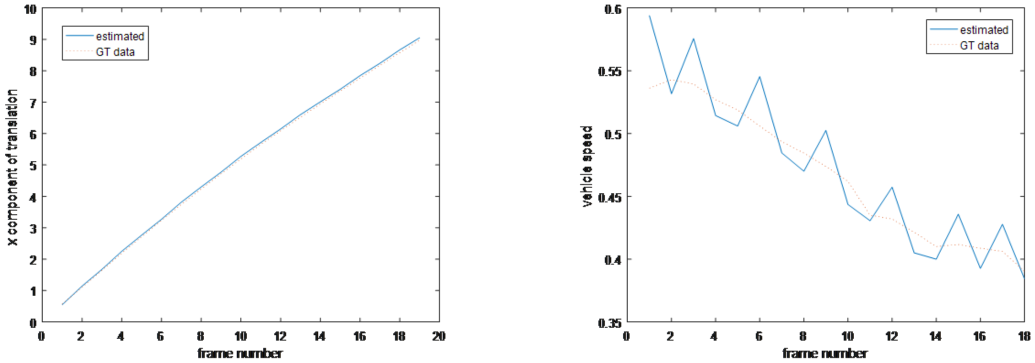


Figure 4.4: X component translation and vehicle speed

4.2 Convolution neural network on depth maps

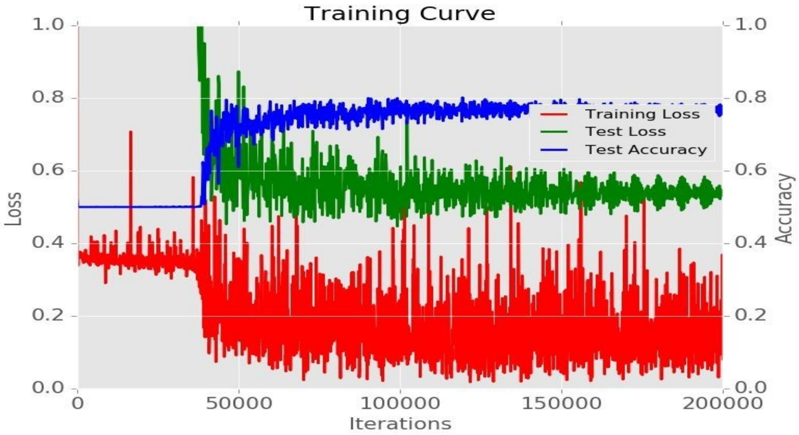


Figure 4.5: Graph of the results of training the CNN

Our baseline method is Histogram of oriented gradient as features plus linear support vector machine as the classifier. The histogram of oriented gradients (HOG) is a feature descriptor used in computer vision and image processing for the purpose of object detection. The technique counts occurrences of gradient orientation in localized portions of an image. It is similar to that of edge orientation histogram.

In figure 4.4, we give the plot of three quantities that defines our training and testing experiments. The Training loss starts decreasing at the 50000 iteration and stabilize to a value between 0.2 and 0.35. Another curve is the testing loss curve, that is calculated on the testing data (which is never seen before data for our model) , the testing loss stabilize between 0.5 and 0.6. The most important curve that represents the accuracy of our model, is the testing accuracy. It defines the ratio of correctly classified data; the accuracy as shown in the table below is at 0.79.

Experiment	Accuracy
Baseline: HOG with L-SVM	0.77
CNN	0.79

Our baseline method trained on our depth map generated data and tested on our testing data is of 0.79 as shown in the result table.

4.3 Lidar and GPS data fusion

As we saw in the previous chapter, the fusion equation depends on the weight a which in terms depends on a constant C that ensures that a < 1.

$$a = (\sigma_L(X)/\sigma_G(X)) * C; \quad C < max(\sigma_G(X)/\sigma_L(X)) \tag{4.1}$$

We experimented with different values of C and found that C=0.3 is the value that result in fusion error less than individual sensor errors as shown in the graph below.

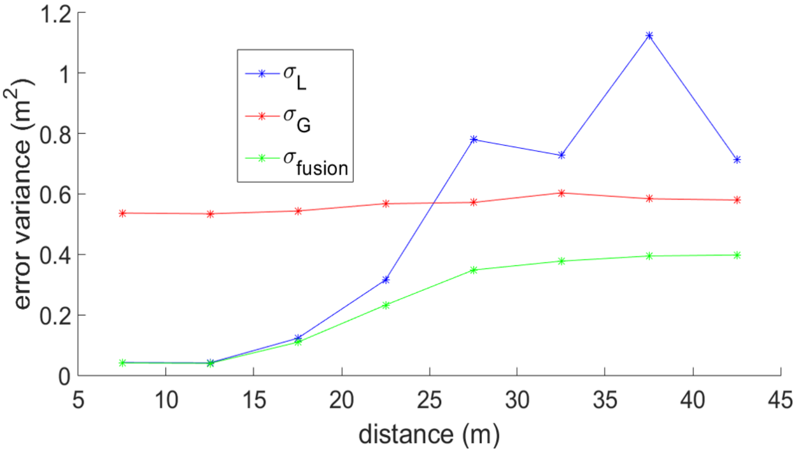


Figure 4.6: Graph of the results of the fusion algorithm - σ_L from experiment and σ_G from simulation

We can see from the above graph that the fusion gives a lower error variance. When either sensor determines lack of accuracy, fusion should not be used; e.g., with low GPS satellite count <5

Fusion can also be done at the level of vehicle tracking (BSM info filtered into tracks). This requires understanding the distance dependent error of V2V based position. Such error depends on many factors: channel congestion, fading, vehicle dynamic, etc. With a known error profile, the same adaptive method above will work with the V2V data, since the weight is distance dependent.

Chapter 5

Conclusion

We have presented a robust EM-ICP algorithm for reconstructing the vehicle environment from a collection of point clouds. Moving objects including vehicles, pedestrians and cyclists are treated as outliers and can be interpreted as the byproduct of reconstruction. Our preliminary experimental results have shown successful reconstruction of vehicle environments for a variety of situations (urban and rural, road and intersection). Along with this line of research, we were able to fuse the scene information extracted from LIDAR data with other sensors (e.g., video imagery). We also presented a linear sensor fusion method and we have shown that it performs better than individual sensors.

A multimodal approach toward pedestrian detection might offer improved performance over any single sensor. Future work may also includes efficient representation of 3D vehicle environment (e.g., octomap-based [24]) and communication of such information among connected vehicles.

Concerning our fusion method, we showed that even a simple fusion methods can provide significant benefits. The challenges we face are as follow; we need the Knowledge of the precision of different methods. Such knowledge may be derived a priori, but sometimes it needs to be determined in real time (e.g., use DOP ¡ 2 or satellite count of GPS, using PER in V2V, or LIDAR SNR) also object matching and synchronization of data in both time and space is necessary. Future work may include the study of other fusion mechanisms and adaptive adjustment of fusion weights using accuracy indicators.

References

[1] J. H. Joung, K. H. An, J. W. Kang, M. J. Chung, and W. Yu, “3d environment reconstruction using modified color icp algorithm by fusion of a camera and a 3d laser range finder,” in *2009 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 3082–3088, Oct 2009.

[2] S. Sivaraman and M. M. Trivedi, “A review of recent developments in vision-based vehicle detection,” in *in IEEE Conf. Intell. Veh. Symp*, 2013.

[3] R. B. Rusu and S. Cousins, “3d is here: Point cloud library (pcl),” in *2011 IEEE International Conference on Robotics and Automation*, pp. 1–4, May 2011.

[4] H. Men, B. Gebre, and K. Pochiraju, “Color point cloud registration with 4d icp algorithm,” in *2011 IEEE International Conference on Robotics and Automation*, pp. 1511–1516, May 2011.

[5] S. Granger and X. Pennec, *Multi-scale EM-ICP: A Fast and Robust Approach for Surface Registration*, pp. 418–432. Berlin, Heidelberg: Springer Berlin Heidelberg, 2002.

[6] J. Hernandez and B. Marcotegui, “Point cloud segmentation towards urban ground modeling,” in *2009 Joint Urban Remote Sensing Event*, pp. 1–5, May 2009.

[7] M. Kusenbach, M. Himmelsbach, and H. Wuensche, “A new geometric 3d lidar feature for model creation and classification of moving objects,” in *2016 IEEE Intelligent Vehicles Symposium, IV 2016, Gotenburg, Sweden, June 19-22, 2016*, pp. 272–278, 2016.

[8] B. Völz, H. Mielenz, R. Siegwart, and J. Nieto, “Predicting pedestrian crossing using quantile regression forests,” in *2016 IEEE Intelligent Vehicles Symposium (IV)*, pp. 426–432, June 2016.

[9] L. E. Navarro-Serment, C. Mertz, and M. Hebert, “Pedestrian detection and tracking using three-dimensional ladar data,” *Int. J. Rob. Res.*, vol. 29, pp. 1516–1528, Oct. 2010.

[10] S. El-Halawany, A. Moussa, D. D. Lichti, and N. El-Sheimy, “Detection of road curb from mobile terrestrial laser scanner point cloud,” vol. XXXVIII-5/W12, 09 2012.

[11] C. Fernández, D. F. Llorca, C. Stiller, and M. A. Sotelo, “Curvature-based curb detection method in urban environments using stereo and laser,” in *2015 IEEE Intelligent Vehicles Symposium (IV)*, pp. 579–584, June 2015.

[12] A. Y. Hata, F. S. Osório, and D. F. Wolf, "Robust curb detection and vehicle localization in urban environments," *2014 IEEE Intelligent Vehicles Symposium Proceedings*, pp. 1257–1262, 2014.

[13] M. Vlaminck, H. Q. Luong, W. Goeman, P. Veelaert, and W. Philips, "Towards online mobile mapping using inhomogeneous lidar data," in *2016 IEEE Intelligent Vehicles Symposium (IV)*, pp. 845–850, June 2016.

[14] P. J. Besl and N. D. McKay, "A method for registration of 3-d shapes," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 14, pp. 239–256, Feb 1992.

[15] C. Stauffer and W. E. L. Grimson, "Adaptive background mixture models for real-time tracking.," in *CVPR*, pp. 2246–2252, IEEE Computer Society, 1999.

[16] P. Jenke, M. Wand, M. Bokeloh, A. Schilling, and W. Straßer, "Bayesian point cloud reconstruction," *Computer Graphics Forum*, vol. 25, no. 3, pp. 379–388, 2006.

[17] R. B. Rusu, N. Blodow, and M. Beetz, "Fast point feature histograms (fpfh) for 3d registration," in *Proceedings of the 2009 IEEE International Conference on Robotics and Automation*, ICRA'09, (Piscataway, NJ, USA), pp. 1848–1853, IEEE Press, 2009.

[18] F. Moosmann, O. Pink, and C. Stiller, "Segmentation of 3d lidar data in non-flat urban environments using a local convexity criterion," in *2009 IEEE Intelligent Vehicles Symposium*, pp. 215–220, June 2009.

[19] J. Landa and V. Ondroušek, "Detection of pole-like objects from lidar data," *Procedia - Social and Behavioral Sciences*, vol. 220, no. Supplement C, pp. 226 – 235, 2016. 19th International Conference Enterprise and Competitive Environment 2016.

[20] C. Premebida, J. Carreira, J. Batista, and U. Nunes, "Pedestrian detection combining rgb and dense lidar data," in *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 4112–4117, Sept 2014.

[21] J. Huang and S. You, "Point cloud labeling using 3d convolutional neural network," in *2016 23rd International Conference on Pattern Recognition (ICPR)*, pp. 2670–2675, Dec 2016.

[22] D. Maturana and S. Scherer, "3d convolutional neural networks for landing zone detection from lidar," in *2015 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 3471–3478, May 2015.

[23] R. E. Kalman, "A new approach to linear filtering and prediction problems," *Transactions of the ASME–Journal of Basic Engineering*, vol. 82, no. Series D, pp. 35–45, 1960.

[24] S. Věchet and J. Krejsa, *Sensors Data Fusion via Bayesian Network*, pp. 221–226. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010.

[25] M. E. Conde, S. Cruz, D. M. Muñoz, C. H. Llanos, and E. L. F. Fortaleza, "An efficient data fusion architecture for infrared and ultrasonic sensors, using fpga," in *2013 IEEE 4th Latin American Symposium on Circuits and Systems (LASCAS)*, pp. 1–4, Feb 2013.

[26] A. Hornung, K. M. Wurm, M. Bennewitz, C. Stachniss, and W. Burgard, "Octomap: an efficient probabilistic 3d mapping framework based on octrees," *Autonomous Robots*, vol. 34, pp. 189–206, Apr 2013.

[27] A. Fitzgibbon, "Robust registration of 2d and 3d point sets," *Image and Vision Computing*, vol. 21/14, pp. 1145–1153, January 2003.

[28] D. Chetverikov, D. Stepanov, and P. Krsek, "Robust euclidean alignment of 3d point sets: the trimmed iterative closest point algorithm," *Image and Vision Computing*, vol. 23, no. 3, pp. 299 – 309, 2005.

[29] M. Ester, H. peter Kriegel, J. Sander, and X. Xu, "A density-based algorithm for discovering clusters in large spatial databases with noise," pp. 226–231, AAAI Press, 1996.

[30] G. Pandey, J. R. Mcbride, and R. M. Eustice, "Ford campus vision and lidar data set," *Int. J. Rob. Res.*, vol. 30, pp. 1543–1552, Nov. 2011.

[31] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun, "Vision meets robotics: The kitti dataset," *Int. J. Rob. Res.*, vol. 32, pp. 1231–1237, Sept. 2013.

[32] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell, "Caffe: Convolutional architecture for fast feature embedding," *arXiv preprint arXiv:1408.5093*, 2014.

[33] L. Wald, "A European proposal for terms of reference in data fusion," in *Commission VII Symposium "Resource and Environmental Monitoring"*, vol. XXXII, (Budapest, Hungary), pp. 651–654, Sept. 1998.

[34] B. V. Dasarathy, "Information fusion – what, where, why, when, and how?," vol. 2, pp. 75–76, 06 2001.

[35] E. Bosse, J. Roy, and D. Grenier, "Data fusion concepts applied to a suite of dissimilar sensors," in *Proceedings of 1996 Canadian Conference on Electrical and Computer Engineering*, vol. 2, pp. 692–695 vol.2, May 1996.

[36] B. V. Dasarathy, "Information fusion – what, where, why, when, and how?," vol. 2, pp. 75–76, 06 2001.

[37] P. J. Nahin and J. L. Pokoski, "Nctr plus sensor fusion equals iffn or can two plus two equal five?," *IEEE Transactions on Aerospace and Electronic Systems*, vol. AES-16, pp. 320–337, May 1980.

[38]

[39] H. F. Durrant-Whyte, B. Y. S. Rao, and H. Hu, "Toward a fully decentralized architecture for multi-sensor data fusion," in *Proceedings., IEEE International Conference on Robotics and Automation*, pp. 1331–1336 vol.2, May 1990.

[40] P. Cheeseman and J. Stutz, "Bayesian classification(autoclass):theory and results," 1996.

[41] A. Elfes, "A sonar-based mapping and navigation system," in *Proceedings. 1986 IEEE International Conference on Robotics and Automation*, vol. 3, pp. 1151–1156, Apr 1986.

[42] R. R. Brooks and S. S. Iyengar, *Multi-sensor Fusion: Fundamentals and Applications with Software*. Upper Saddle River, NJ, USA: Prentice-Hall, Inc., 1998.

[43] L. Plotkin, "Pydriver: Entwicklung eines frameworks fur r aumliche detektion und klassifikation von objekten in fahrzeugumgebung," *Bachelor's thesis (Studienarbeit), Karlsruhe Institute of Technology, Germany*, 2015.

[44] S. Salti, F. Tombari, and L. D. Stefano, "Shot: Unique signatures of histograms for surface and texture description," *Computer Vision and Image Understanding*, vol. 125, no. Supplement C, pp. 251 – 264, 2014.

[45] S. Rusinkiewicz and M. Levoy, "Efficient variants of the icp algorithm," in *Proceedings Third International Conference on 3-D Digital Imaging and Modeling*, pp. 145–152, 2001.

West Virginia University Electronic Thesis and Dissertation Signature Form

Student Name: Cheikh Sidiya (Last) Ahmed (First) (Middle)

Student ID #: 800161499

Non-WVU Email Account: cheikhsidiyahmed@gmail.com

Degree: x Master's ___ Doctorate

Document Type: x Thesis ___ Dissertation

Document Title: Point Cloud Processing Algorithms for Environment Understanding in Intelligent Vehicle Applications

Student Agreement:

I hereby certify that, if appropriate, I have obtained and attached hereto a written permission statement from the owners of each third party copyrighted matter to be included in my thesis, dissertation, project report, or other research material, allowing distribution as specified upon deposit.

I hereby grant to West Virginia University and its agents the non-exclusive license to archive and make accessible, under the conditions selected upon deposit, my above mentioned document in whole or in part in all forms of media, now or hereafter known. I retain ownership rights as specified in the WVU copyright policy to the copyright of the abovementioned document. I also retain the right to use in future works (such as articles or books) all or part of this abovementioned document.

Review and Acceptance:

The above mentioned document has been reviewed and accepted by the student's advisory committee. The undersigned agree to abide by the statements above, and agree that this Signature Form updates any and all previous Signature Forms submitted heretofore.

Signed: Ahmed Sidiya (Student) 11/29/2017 (date)

Committee: (Committee Chair) 11/29/2017 (date)

(Committee Member) 11/29/2017 (date)

Yaser P Fallah (Committee Member) 11/29/17 (date)

(Committee Member) (date)

(Committee Member) (date)

(Committee Member) (date)

Scanned by CamScanner