Fusion Sensor for Autonomous Cars

Luis Alejandro Sierra Polanco
Student Id: 864153

Advisor: Prof. Matteo Matteucci

Co-Advisor: Simone Mentasti

Dipartimento di Elettronica, Informazione e Bioingegneria
Politecnico di Milano

This thesis is submitted for the degree of
Master of Science in Computer Science and Engineering

Academic Year 2017-2018

I would like to dedicate this thesis to all the Pandas around the world. . .

Acknowledgements

I would like to thank,

My dear family who without hesitation has been giving me a hand, supporting me throughout all my education and life, especially my mom who made all this possible, also my brother, my dad and my uncle, because without their support this journey would had been much more harder.

To my friends back in Colombia that even from the distance have been always supporting me and also to my most recent friends, those who I met along this journey who not only cheered me up, but also helped me to build amazing memories inside and outside the university.

To my university, Politecnico di Milano, who gave me the chance to have this experience and get not only all the knowledge to expand my field, but also this valuable degree.

Abstract

An autonomous vehicle is a complex machine which recollects information and researchers in different fields, from those researching the ethical consideration and the impact that a service like this could have in the society, to all the engineers improving the design, hardware and software of it, with the objective of build an autonomous car which give the best performance.

This work studies the autonomous car and it goes deeper in the topic of fusion sensor and specifically its use in the generation of the occupancy grid for the object detecting system, and ground removal, in order to avoid false positives and consider the ground as an object.

Different structures and algorithms were considered for this study. The input data used was taken from sensors present in the vehicle.

Abstract

Un'auto a guida autonoma é una complessa machina che raccoglie informazioni e ricercatori in diversi campi, da quelli che ricercano le considerazioni etiche e l'impatto che un servizio come questo avrebbe sulla societá, fino agli ingegneri che cercano di migliorare il disegno, l'hardware ed il software con l'obbiettivo di creare un auto a guida autonoma con le migliori prestazioni.

Questa tesi consiste nello studio dell'auto a guida autonoma riguardo la fusione di sensore, nello specifico sul suo uso nella creazione della "occupancy grid" per il sistema del riconoscere oggetti, e la rimozione della terra al fine di evitare dei falsi positivi e considerare la terra come un ostacolo.

Diverse strutture e algoritmi sono state considerate per queste studio. I dati usati somo raccolti dai sensori presenti nella macchina.

Contents

List of Figures

Chapter 1

Introduction

Fully Autonomous driving in real urban setting has been receiving a great attention. During the past years the technology growth and improvement had made possible to do notable attempts to build an autonomous car, where this attempts allowed us to reach important milestones, that derived in the creation of different vehicles which has different levels of autonomy.

Giving the tools and the different milestones reached it has being able to establish a common definition for automated driving in order to simplify communication. This definitions lead to establish the autonomy levels which are characterized by the amount of autonomy the vehicle has, this description was made by the SAE international (Society of Automotive Engineers) [10] giving us 6 levels of autonomy ranging from no driving automation, level 0, to a full driving automation, level 5.

This project is focus on the research of the autonomous car, its parts and functioning, with a special focus on object detection and ground removal where the implementation was done considering different approaches, based on the data captured by a the sensors installed in the autonomous vehicle prototype presented in the chapter 4, subsection 4.1.

The first steps were a review of the state of the art of the autonomous driving in order to understand it and divide it in parts to end up with a focus on a specific section, through meetings we realized the dimension of the project of an autonomous vehicle, this allow us in setting the goals of the research in collecting information and begin the implementations and improvements in the object detection and the ground removal leaving the rest for ongoing and future works as described in chapter 5. The structure is as follows:

- Chapter 2. State of Art defines and explains the concepts, structure and needed background knowledge of an autonomous car.
- Chapter 3. Related work, presents previous work related to this thesis.
- Chapter 4. Development, the description of the results from a theoretical and technical point of view.
- Chapter 5. Conclusion and future work, summarizes the improvements done and describes future work.

Chapter 2

State of the art

2.1 Autonomous vehicle

2.1.1 Definition

An autonomous vehicle its a type of robot, this study case is about terrestrial vehicle and more specifically about autonomous cars which can be also known as driver-less car. This kind of robot is a vehicle which is capable of sensing its surrounding and move with little or no human intervention, following the traffic rules and optimized path avoiding the possible obstacles of its environment.

The autonomous cars are capable to combine a variety of sensors to perceive their environment, such as radar, LIDAR, sonar, GPS and inertial measurement units (IMU), among others. It also has an advanced control system to interpret the data from the sensors in order to define its ego-position and execute the object detection to search the possible obstacles and consider relevant signage in order to identify the appropriate navigation paths.

2.1.2 Architecture

Since the autonomous car is a type of robot, different approaches involved in the cognitive architecture as the Sense-Plan-Act model were considered. Based on the previous model, the first section, sense, is where this project take place. The first section will contains all the

sensors and its information, and once acquired is processed to generate a representation of the vehicle's environment so it can understand it and move through it.

In order to follow an architecture which has the required performance for an autonomous car, it needs a mental model. The base to build the system and the three main mental models of the classical cognitive approach are:

- Deliberative.
- Reactive.
- Hybrid.

The base chosen for the system was the Hybrid which combine the best of the deliberative and the reactive approaches in a single architecture, taking the representation, the models and the planning from the deliberative approach using it as a strategic planning and reasoning. It implements the reactive approach as a low level control and behaviour for real-time response, a multiple goals selection giving the system a more robust, flexible and with modularity implementation.

With this said it can be established that the autonomous car control software consists in four functional subsystems which are the following [4]:

- Perception subsystem
  - It collect the available information about the car's road traffic environment, to manage and process it.
- Real-time decision making and driving maneuver control.
  - It is in charge of making the different driving decisions.
- Driving maneuvers
  - It contains a set of closed-loop control algorithms in order to be able to maneuver the vehicle in a specific traffic situation.

2.2 Perception subsystem

2.2.1 Sense structure

The sense structure is implemented with the goal of generate the optimal sensor fusion in order to produce an accurate understanding of the surroundings of the car.

The autonomous car is capable to sense its environment due to a control system which interpret the information coming from a combination of a variety of sensors such as Lidar, radar, sonar, cameras, inertial measurement units (IMU) and Global Positioning System (GPS). The interpretation made by the control system will give the required tools to identify the obstacles, the relevant traffic signs and an appropriate navigation path.

Even if there are present two main sections in the sense architecture, the navigation and object detection, the best result will be given by a union between both section so their result can be more accurate.

2.2.2 Sensor fusion

Sensor fusion also known as multi-sensor data fusion, is a process in which the system combines intelligently the data coming from several sensors to improve the accuracy and performance of individual sensors, generating useful information and correcting the deficiencies of single sensor allowing to calculate more accurately the ego-position-detecting of the vehicle and the object-detecting of its environment [13].

This process can be divided in three main parts [1] which are categorized based on the abstraction level where the fusion data takes place, the parts are:

- Low-level (early).
  - It is the fusion which combines sensor data to create a new set of data
- Mid-level
  - It is the fusion that integrates features.
- High-level (late or decision-level)
  - It is the fusion that combines the classified outputs.

2.2.3 Positioning

The positioning of the vehicle, also known as localization and navigation is the process in which it is determined where the vehicle is located with respect to its environment and the world model. Positioning is one of the most fundamental processes required by an autonomous car because it gives the knowledge of its own location and with this information he is capable to design and implement a path, making the correct decision in order to navigate and reach its destination through its environment.

Some of the ego-position-detecting sensors used in the literature with some characteristics are:

- GPS (Global positioning System):
  - Good in long-term accuracy but poor in short-term accuracy.
    - Is influenced by the jamming in the environment and its signal can be obstructed
  - Give the absolute position
- INS (Inertial Navigation System):
  - Good in short-term accuracy but poor in long-term accuracy
  - Its main Component is the IMU (Inertial Measurement Unit) that measure:
    - Velocity
    - Orientation
    - Gravitational forces
  - It is a sensor that give the position with coordinates x, y and the yaw angle.
- GPS/INS: Integration of the two sensors will support each other
  - INS errors:
    - The rate gyro bias error from the gyroscope measurement.
    - The bias errors from the accelerometer measurements
    - Small errors in measurements of IMU are integrated into larger error in the velocity and orientation generation an accumulated error.
  - To correct them the information of the GPS and its correct position data are used

  - GPS errors:
    * The precision in the localization due to interference of the signal and jamming with the environment.
  - This are correct with the information giving by the INS sensors and using the process Dead Reckoning.

- Stereo Vision Sensor
  - Recognizes and tracks the lane marks
    * 3D geometry of the line
    * Position of the vehicle relative to the line
  - Recognizes the traffic signs.
  - Recognizes the traffic lights.

- Vehicle sensors
  - Velocity of the rear axis
  - Steering angle

In order to fulfill the proper ego-position-detecting data fusion different algorithms and methods had been developed and implemented through the years, including a combined information with the object-detecting sensors information, to do the estimation of the absolute vehicle position (x,y) and the orientation Ψ in the world coordinates, not leaving behind the object movement evaluation which improves the estimation of the velocity and angular yaw, by considering information about the relative motion objects in the vehicle environment.

Some of the methods and algorithms used for the navigation data fusion are:

- Dead reckoning
  - It is used to estimate the current vehicle position based on the previous known position, its known or estimated velocity over a elapsed time and its yaw angle.

- Kalman filter
  - Applicable only for linear systems, but combining it with fuzzy logic is a good way to do detection and correction of divergence of Kalman filter.

  - The purpose of the fuzzy logic adaptive system (FLAS) is to detect the bias of measurements and prevent divergence of the Kalman filter.

- Extended kalman filter

  - Applicable to non linear systems. It is used to estimated the heading angle and position of the vehicle to navigate from one location to another. The position of the vehicle is obtained from the GPS/INS.

- Interval analysis method

  - It is used for the estimation of the heading angle and the x and y position of the vehicle in interval.
    * Its main advantage is that guarantees on the bounds that makes the system less sensitive to the problem of consistency of typical filters such as the EKF.

- Covariance intersection algorithm

  - It is a data fusion algorithm which has a convex combination of the means and the covariance in the information space.
    * The position and direction information are obtained from the information provided by the resulting covariance matrix of the EKF estimates.

- Artificial neural networks

  - Used to give a solution to facilitate measurement fusion in decentralized architectures where either the source of information or the sensor noise is unknown.
  - Implemented to estimate feature uncertainty from data, without previous knowledge of the sensors characteristics.
  - Recommendable particularly for highly non linear applications involving larger number of parameters.
  - The main benefit of using Artificial Neural Networks resides in the capability of approximating a process or set of functions provided that the network structure and size is sufficiently large.
  - Feedback Neural Networks (FBNN) are able to capture possible relationships between current and past signals.
  - Data analysis

        * To avoid measuring the relative influence of the signal a double analysis of the input set is done, one before and one after the Neural Networks training.
    - Two algorithms are used for training to compensate for deficiencies of the so-called error backpropagation algorithm (EBP), this algorithms are:
        * Levenberg-Marquardt (LM).
        * Scaled Conjugate Gradient (SCG).

2.2.4 Object detection

The object detection is the computer technology in charge of the data processing, which allows to deal with the detection of different instances of semantic objects, that are a part in certain class as for example, in this topic, the autonomous cars, there are objects which must be detected as pedestrians, other vehicles, static objects or any other possible obstacle that can affect the trajectory of the vehicle, producing damages or injuring another living being beside its users.

Some of the object-detecting sensors with their characteristics used in the literature are:

- Stereo Vision
    - Performs the track of the object detection.
    - lane recognition.
    - Traffic signals recognition.
    - To build the Stixel-world[9], a compact medium level representation of the 3D-world can be used by implementing a 5 step process.
- Laser scanners
    - Different kind of laser scanners can be used.
        * Single beam laser
        * Triple beam laser: It is more robust to dynamic disturbances than single beam lasers
- Long range RADAR sensors
- RADAR

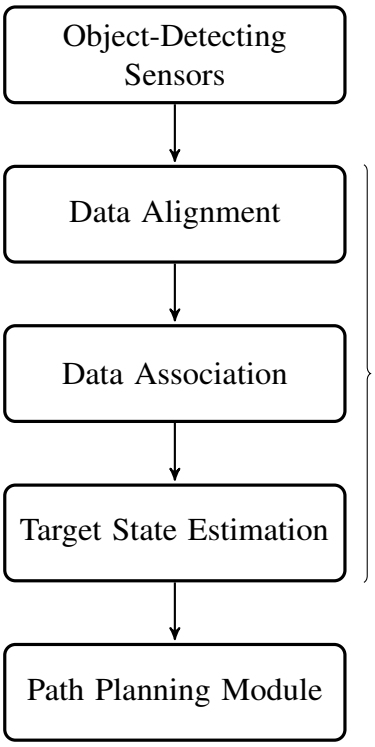


Figure 2.1 Structure of the sensor fusion module

- LIDAR

The object detecting fusion data process is composed by three main stages where the signal is prepared, organize and processed.(see Fig. 2.1).

The process of object detecting should organize and pre-process the information coming from the different sensors, since this are dissimilar, work independently and unsynchronized from each other, and the object tracking is perform individually on each of the sensors. All the data arriving from the different sensors has to be synchronized before other processes, so it can be align and ready for any further manipulation, this section is called data alignment.

The data set of the sensors consist in position and kinematic variables where each sensor use target description which fit to the measurements of the objects, this target descriptions are define by the type of the sensor and algorithm used.

The object representation needs to have a compact representation, because it is necessary to guarantee a real-time communication between the sensors and the fusion system, having this in mind the object representation is established. For the vision systems which are based on cameras describe the objects as a cuboid that surrounds the detected target with its sides

parallel to the axes of the sensor, for another hand with lase-scanner sensors it is used to describe an object by three characteristics points which consist of the point of the detected object closed to the sensor as well as of the outermost points to the left and right of the object.

Following with the process the next step is data association and it is in charge to do a correlation of the observations from the sensors in order to identify which ones represent the same object.

The final step in the process is the target estimation or track data fusion. It is when the objects that are the same in different sensors are combined giving them a unique representation.

Due to the variety of sensors and their different type of data, the methods and algorithms of the object detection are specific related to the sensors or to a part in the process. Some of the algorithms used in the literature are the Semi-Global Matching (SGM) which it is used for stereo vision sensor, as it can be seen implemented in the Stixel-world [9]. In Data Association the preference is to use the Nearest-Neighbor in order to satisfy and guaranty the real-time requirements of the system.

The Kalman filter algorithm can be well used during the process since it is used to reduce the influence of the measurement noise, to estimate the state vector, in the target state estimation and for the data fusion where the state vector modified and updated by using the measurement from two different sensor who has an observation of the same object, for this the State-Vector-Fusion algorithm is used since takes into account the effects of the cross-correlation of the measurement data in the two sensors.

The object detecting system should consider the next five sections inside its process in order to improve its recognition of the different object and avoid to register a non-obstacle as one:

- Sensors and Ego motion estimation
- Ground Surface estimation.
- Representation for on-ground obstacle detection.
- Motion detection.
- Data association

Sensors and Ego motion estimation

As it is mentioned, the common sensors used in this section are Stereo cameras, Velodyne LIDAR, 2D LIDAR and Radar. The ego-motion estimation is used to transform the sensor data from the ego-vehicle to the world coordinate system, in order to realize this transformation different methods can be used as for example, Global Navigation Satellite System (GNSS), GPS/IMU odometry, Visual odometry and scan matching.

Ground surface estimation

The ground surface estimation it is an important section because its estimation is used as a ground base for the representation for on-ground obstacle detection and to avoid being taken it as an obstacle due to the sensor data received,. For this estimation the assumption of a planar ground is taken by means of algorithms as RANSAC which uses a region growing and least square fitting for this goal, also it is used a low average and variance in height of the cells that contain points.

Representation for on-ground obstacle detection

For the representation of the on-ground obstacles it is needed to establish a type of representation of the world, this representation depends on the goal, the sensors and the data that its being treated, for this they go from a 2D representation to a 3D representation where in some situation may contain more information than the 3D position, thanks to the use of different layers.

Some of the representation used for the on-ground obstacle detection are:

- 2D Occupancy/Velocity grid
  - The occupancy grid [8] [11] is a multi-dimensional random field model that maintains probabilistic estimates of the occupancy state of each cell in a spatial lattice. Bayesian estimation mechanisms employing stochastic sensor models allow incremental updating of the Occupancy Grid using multi-view, multi-sensor data, composition of multiple maps, decision-making, and incorporation of robot and sensor position uncertainty.

  - The velocity grid [15] is the estimated velocity distribution for each cell of a grid. It is based on a prediction-estimation paradigm, As an input, it uses an observed occupancy grid. On its output, it provides an estimated occupancy grid as well as a set of velocity grids, representing the probability distribution over possible velocities for each cell.

- 2.5D digital elevation map (DEM)
- 2.5D elevation grid
  - The 2.5D grid [2] stores in each cell of a discrete grid the height of objects above the ground level at the corresponding point of the environment. For a vehicle moving on a single underlying surface, a 2.5D grid provides sufficient information to represent the surrounding environment.
- Stixel
  - The Stixel World [12] is a representation of the sensory data that provides compressed and structured access to all relevant visual content of the scene.
  - It is a medium-level model of the environment that is specifically designed to compress information about obstacles by leveraging the typical layout of outdoor traffic scenes.
- 3D voxel grid
- VoxelNet
  - VoxelNet [18] divides a point cloud into equally spaced 3D voxels and transforms a group of points within each voxel into a unified feature representation through the newly introduced voxel feature encoding (VFE) layer.
- Octomap
  - The OctoMap [5] is a 3D occupancy grid mapping approach, provide data structures and mapping algorithms where the map implementation is based on an octree and is designed to be a full 3D model, to be updatable, flexible and compact.

Motion detection

- The Motion detection is the section on the process in which uses the local grid map and the 2.5D grid to generate a 2.5D motion grid and spatial reasoning to suppress false detection. It is also in charge to fill the holes and compensate the velodyne scan's gaps with the use of morphological operators.

Data association

- The data association can use different strategies in order to do a correlation of the observations to determine which detected object goes with which track. which are:
  - Kalman filter.
  - Greedy approach based on a distance function.
  - Nearest neighbor.
  - Global nearest neighborhood (GNN).

Chapter 3

Related Work

3.1 A universal Grid Map

The universal Grid map [14] is a mapping framework for mobile robotics. Its applications include online surface reconstruction and terrain interpretation for rough terrain navigation. Some of the characteristics of the mapping framework are the capacity of creating multy-layered maps, computationally efficient responding of the map boundaries and compatibility with existing ROS map message types. The characteristic of being multy-layered give the advantage of supporting multiple data layers, for example applicable to elevation, variance, color, surface normal, occupancy etc. A multy-layered representation can be see in Fig. 3.1

The algorithms typically developed for ground robots which were designed to move on flat terrain are based on a two-dimensional abstraction of the environment, but when we talk

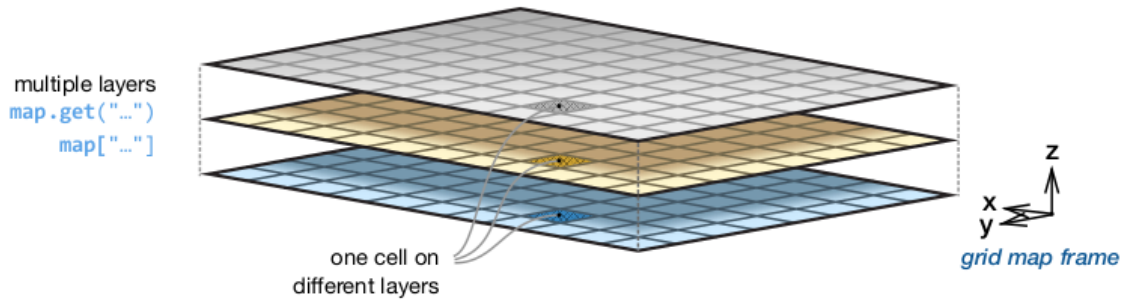


Figure 3.1 The grid map library uses multilayered grid maps to store data for different types of information. [14].

about navigation in rough terrain, the algorithms must be extended to take into account all three dimensions of the environment. This can be done with a 2.5-dimensional representation, where each coordinate on the horizontal plane is associated with an elevation/height value.

The universality in the library, and in relation of the Grid Map is in the sense of not having a restriction to any special type of input data or processing step. The development of the universal grid map library is for being use as a generic mapping framework for mobile robotics with the Robotic Operating System (ROS).

With the objective of giving the robots the opportunity to navigate in previously unseen, rough terrains, it requires to reconstruct the terrain as the robot moves through the environment, In order to do it the system needs a robot pose estimation which in many cases for the autonomous robots, the pose estimation is prone to drift, but this generates an issue, stitching fresh scans with previous data leads to inconsistent maps which as a solution, it was formulated a probabilistic elevation mapping process from a robot-centric perspective. The method consist of three main steps:

- Measurement Update:
  - By means of the kalman filter new measurements from the range sensor are fused with existing data in the map.
- Map Update:
  - For a robot-centric formulation, the elevation map needs to be updated when the robot moves. the changes of pose covariance matrix to the spatial uncertainty of the cells in the grid map are propagated and this reflects errors of the pose estimate in the elevation map.
- Map Fusion:
  - When map data is required for further processing in the planning step, an estimation of the cell heights is computed. This requires to infer the elevation and variance for each cell from its surrounding cells.

When this method is applied, the terrain is reconstructed under consideration of the range sensor errors and the robots pose uncertainty. The information about the height estimate and corresponding variance are store in each cell of the grid map. The region ahead of the sensor has typically the highest precision as it is constantly updated with new measurements, but

the regions out of the sensor's field view have decreased certainty due to drift of the robot's relative pose estimation.

3.2 The Point Cloud Library (PCL)

The Point Cloud Library (PCL) [16] is a standalone, large scale, open project for 2D/3D image and point cloud processing.

PCL is divided in a number of modular libraries. The most important set of released PCL modules are:

- **Filters:** The Filters library contains a set of algorithms for cleaning and processing the signal, it also interacts with other modules as Sample Consensus, kdtree and octree.
- **Features:** The features library contains data structures and mechanisms for 3D feature estimation from point cloud data.
- **Keypoints:** The keypoints library contains implementations of two point cloud keypoint detection algorithms. Keypoints are points in an image or point cloud that are stable, distinctive, and can be identified using a well-defined detection criterion.
- **Registration:** The registration technique is used for the combination of several datasets into a global consistent model. The main goal is to identify corresponding points between the data sets and find a transformation that minimizes the distance (alignment error) between corresponding points.
- **KdTree:** The kdtree library provides the kd-tree data-structure, using FLANN, that allows for fast nearest neighbor searches.
- **Octree:** The octree library provides efficient methods for creating a hierarchical tree data structure from point cloud data.
- **Segmentation:** The segmentation library contains algorithms for segmenting a point cloud into distinct clusters.
- **Sample Consensus:** The sample consensus library holds Sample Consensus (SAC) methods like RANSAC and models like planes and cylinders. These can combined freely in order to detect specific models and their parameters in point clouds.

- **Surface:** The surface library deals with reconstructing the original surfaces from 3D scans. Depending on the task at hand, this can be for example the hull, a mesh representation or a smoothed/resampled surface with normals.
- **Range Image:** The range image library contains two classes for representing and working with range images. With knowledge of the camera's intrinsic calibration parameters, a range image can be converted into a point cloud.
- **I/O:** The I/O library contains classes and functions for reading and writing point cloud data (PCD) files, as well as capturing point clouds from a variety of sensing devices.
- **Visualization:** The visualization library was built for the purpose of being able to quickly prototype and visualize the results of algorithms operating on 3D point cloud data.
- **Common:** The common library contains the common data structures and methods used by the majority of PCL libraries. The core data structures include the PointCloud class and a multitude of point types that are used to represent points, surface normals, RGB color values, feature descriptors, etc. It also contains numerous functions for computing distances/norms, means and covariances, angular conversions, geometric transformations, among others.
- **Search:** The search library provides methods for searching for nearest neighbors using different data structures, including, KdTree, Octree, brute force and specialized search for organized datasets.

3.3 The Stixel World

The Stixel world [9] [12] is the representation of the world produced by the segmentation of an image into superpixels, where each superpixel is a thin stick-like segment with a class label and a 3D planar depth model.

The Stixel world is based on the unsupervised bottom-up segmentation, on road scene models and on semantic segmentation. The bottom-up segmentation aims to partition an image into regions of coherent color or texture. In the road scene models the occupancy grid maps are a predominant role representing the surrounding of the vehicle and the semantic segmentation makes use of deep convolutional neural networks in particular fully convolutional networks (FCNs) for segmentation and classification performance, this segmentation

only model one-dimensional to strike a good balance between accuracy and computational complexity in view of real-time automotive vision.

During the process building the stixel world the image segments are classified into three main categories, where each of them are further subdivided into semantic classes, the main three categories are the support, the vertical and the sky.

The support category has a planar assumption and the stixels are parallel to the ground at a constant high. This category contains the ground, roads and sidewalks among others.

The vertical category has the stixels perpendicular to the ground because in this category it can be found the objects and elements that are on the support class, for example includes the buildings, trees, other vehicles, pedestrians, bicyclist, obstacles among others. Finally the sky category has also the stixels perpendicular to the ground but it takes care to everything which is in the upper part of the image and it is above the objects and elements of the vertical category.

The stixel segmentation is obtained by minimizing the energy function and it uses four metrics. The first one is the Depth accuracy that is define as the percentage of the disparity estimates that are considered inliers. The second one is the Semantic performance which does the average intersection-over-union (IoU) over all classes. The third metric is the Runtime that access the complexity of the obtained representation and the last metric is the number of stixels per image, the third and fourth metric are used as a proxy to assess the complexity of the obtained representation.

The base line used was called smart down-sampling, this because the regular down-sampling of the disparity image would result in very strong blocking artifacts and this in particular on the ground plane. Instead of the regular down-sampling the pixel-level semantic input is used to differentiate three depth models which are the analog of the already mentioned three structural classes used in the stixels. Following this analog representation, we have the ground pixels where there is assign the mean deviation to the flat ground hypothesis, for the pixels covering the vertical obstacles the assignment is the mean disparity and the last structure is the sky pixels in which the assignment is the disparity zero.

3.4 Making Bertha see

Bertha is a fully autonomous Mercedes Benz S-class [17] that was used to drive autonomously from Mannheim to Pforzheim, Germany, a distance of 100 km. It used the Stixel world as the underlying visual environment representation and the sensors used in this project were:

- Four 120° mid-range radars
- Stereo Camera
- Two wide angle monocular color cameras
- GPS and Inertial vehicle sensor

The vision system of the car was composed by one wide-angular monocular camera used for the traffic light recognition and the pedestrian in turning maneuvers, one wide-angle monocular camera for the feature based localization and finally a stereo system which is used for the 3D scene analysis and the lane recognition, with this last one three issues were found, the first is that there are no strict rules applied for urban roads, the second is related to the low speed in which allow foe rapid changes in a line course, finally the lack of marked lanes or the fact that they are marked sparsely produce error during its recognition.

The pipeline of the stereo vision is composed by four steps.

- Dense stereo reconstruction: the dense stereo reconstruction itself it is in charge of the stereo matching where a dense disparity images are reconstructed using semi-global matching (SGM).
- The stixel segmentation.
- Motion estimation of other object: It process the data to do a detection and a tracking of other moving traffic participants which is achieved by tracking stixels over time using kalman filtering.
- The final object segmentation.

In order to realize a proper pedestrian recognition it needs to have a real-time performance, in this vehicle a vision-based pedestrian detection system consist on three main modules, which are:

- Region-of-interest (ROI) generation: As the name says it, it generate the region of interest of the data where involves the recovery scene geometry in terms of camera parameters and 2D road profile for dense stereo vision.
- Pedestrian classification: This classification is done using linear support vector machines.
  - Tracking: The tracking employ a standard recursive Bayesian formulation involving Extended kalman filters (EKF) with an underlying constant velocity model of dynamics.

3.5 Multimodal vehicle detection

In this project proposes a real-time multimodal vehicle detection system integrating data from a 3D-LIDAR and a color camera [1], from this two sensors, three kind of data are extracted, one type from the monocular color camera which is the RGB image and two types from the 3D-LIDAR sensor which are the Dense-Depth Map (DM) that is an up-sampled representation of the sparse LIDAR's range data and the Dense-Reflectance Map (RM) which is a high-resolution map from LIDAR's reflectance data.

The model is divided in three modules, the first one is the Multimodal data generation that contains the sensors and extract the three types of data that are the input of the next module, the second is the individual detection, in this module it is used a Deep ConvNet-based detection framework called YOLO (you only look once) for a real-time object detection done individually for each type of data and finally a third module where the fusion data happens. A representation of the model can be seen in Fig. 3.2.

The multimodal detection fusion system tries to use the associated confidence of individual detection and the detected Bounding Box's (BB) characteristics in each modality to learn a fusion model and deal with detection limitation.

3.6 Approach based on voxels and multi-region ground planes

The system proposed in this work is composed of two main modules:

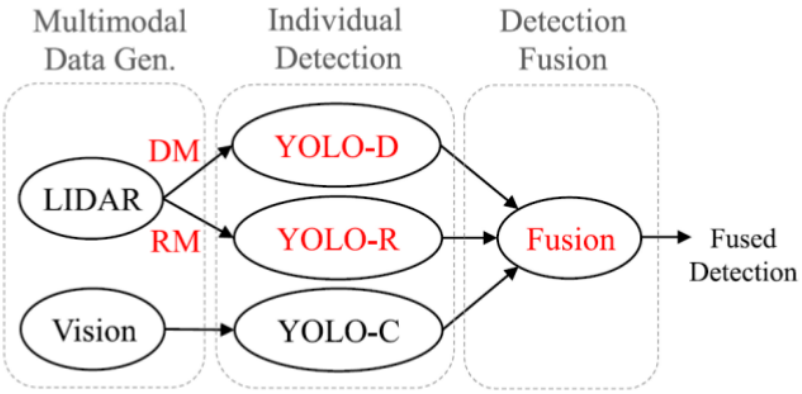


Figure 3.2 The conceptual model of the multimodal vehicle detection.

- An effective ground surface estimation using a piece wise plane fitting algorithm and RANSAC-method.
- A voxel-grid model for static and moving obstacles detection using discriminative analysis and ego-motion information, taking into account that a voxel is 3D occupancy grid composed of equally-sized cubic volumes.

In order to do the ground Surface estimation a dense point-cloud generation is used. The dense point-cloud construction begins by transforming the pointclouds from ego-vehicle to the world coordinate system using GPS/IMU localization data. For the ground Surface estimation a piece wise plane fitting algorithm is used in order to estimate the ground geometry and this algorithm is composed by 4 steps:

- Slicing.
- Gating.
- Plane fitting.
- Validation.

Inside this approach there are three more thing it must be taken into account, the ground/on-ground obstacle separation, the voxelization and the discrimiantive stationary/moving obstacle segmentation.

- Ground/on-ground obstacle separation: The obstacle separation is performed based on the distance between the points inside each slice region to the corresponding surface plane.

- Voxelization: The voxel grids are dense 3D structures with no dependency to predefined features which allow them to provide detailed representation of such complex environments. The voxelization process is perform by essentially two steps, the first one is quantizing end-points of the beams and the second step is computing the occupancy values.
- Discriminative stationary/moving obstacle segmentation: In this part the voxels present on each scan will give the discrimiation between a stationary and a moving object, this discrimination is done thanks to the voxels due to. the stationary objects are mapped into the same voxel in consecutive scans and the moving objects occupies different voxels along time.

3.7 VoxelNet

The PointNet [7] is an End-to-end deep neural network that learns point-wise features directly from point clouds, it is known that have good results on 3D object recognition, 3D part segmentation and in the point-wise semantic segmentation. It works with a Region Proposal Network (RPN) that is a high optimized algorithm for efficient object detection but it present issues with a typical LIDAR point cloud due that requires data to be dense and organized in a tensor structure.

In this project is presented a VoxelNet [18] which scale up the PointNet presented in [7] and its improved version presented in [6] which enable the network to learn local structures at different scales. The VoxelNet architecture is proposed as:

1. Feature Learning network
    - Voxel partition
    - Grouping
    - Random sampling
    - Stacked voxel feature encoding
    - sparse tensor representation
2. Convolutional muddle layers
3. Reagional proposal network

3.8 Most used dataset

The most used dataset for autonomous vehicles are the KITTI, KITTI'15 and CityScapes and their characteristics are:

- KITTI
  - It is the only dataset containing dense semantic labels and Depth ground truth, it was capture with a Velodyne 3D lase scanner a high-precision GPS/IMU inertial navigation system and it contain 11 set of classes, which contains different vehicle situation, object situation, type of obstacle and scene condition.
- KITTI'15
  - In this dataset there is no suitable semantic ground truth, it was capture with a Velodyne HDL-64 laser scanner and it contain 200 images with sparse disparity ground truth.
- CityScapes
  - This is a Highly complex and challenging dataset with dense annotation of 19 classes on 3475 images, in this dataset there are stereo views available but do not exist the ground truth disparities.

Chapter 4

Development

This project started with the idea of an autonomous car, its architecture and requirements, but due to the magnitude of the project it started to be focus in the sensors and world representation of the autonomous car, finalizing in a fusion sensor for autonomous car, based on the object detection and ground removal.

In order to proceed, there are some terms and concepts that should be explained as they are going to be used throughout the rest of this work.

- **costmap 2d:** The costmap 2d is a package implemented in ROS which provides a configurable structure that maintains information about where the robot should navigate in the form of an occupancy grid. The costmap uses sensor data and information from the static map to store and update information about obstacles in the world.
- **The Point Cloud Library (PCL):** It is a standalone, large scale, open project for 2D/3D image and point cloud processing.
- **Random sample consensus (RANSAC):** RANSAC is an iterative method to estimate parameters of a mathematical model from a set of observed data that contains outliers, when outliers are to be accorded no influence on the values of the estimates. It is a non-deterministic algorithm in the sense that it produces a reasonable result only with a certain probability, this probability increases as more iterations are allowed.
- **k-d tree:** It is a space-partitioning data structure for organizing points in a k-dimensional space. k-d trees are a useful data structure for several applications, such

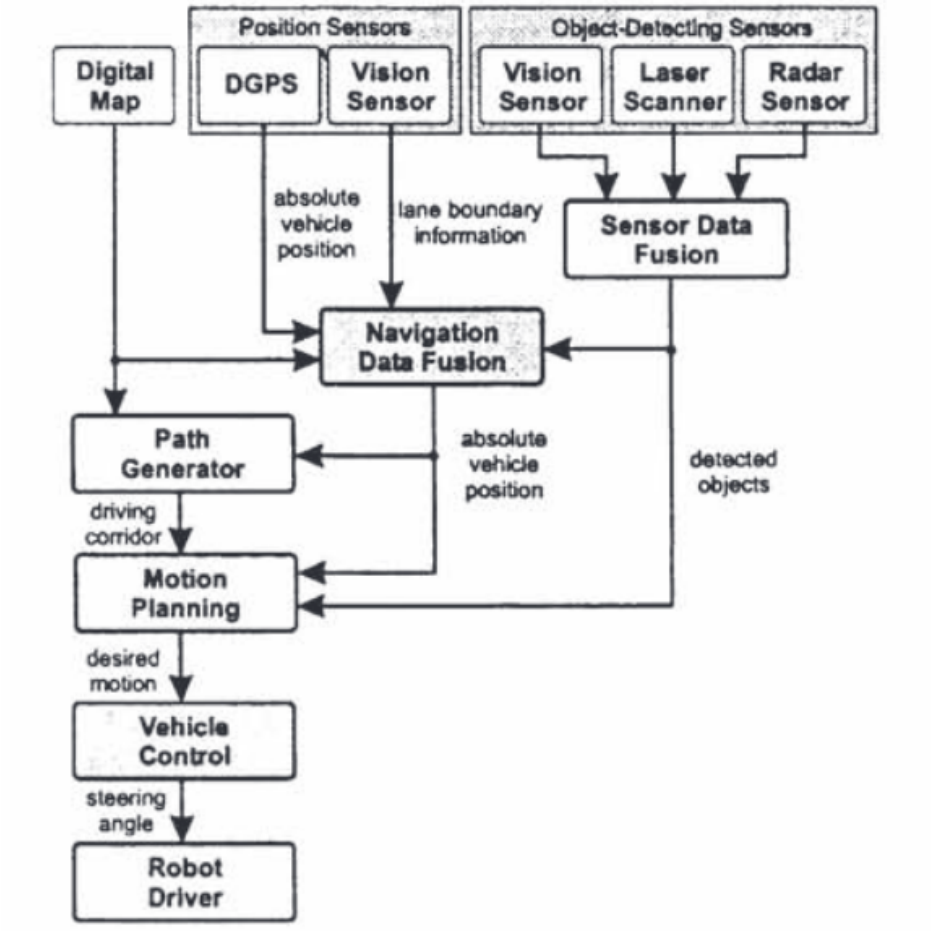


Figure 4.1 Structure of the vehicle guidance system

as searches involving a multidimensional search key as for example range searches and nearest neighbor searches.

- **Ego-position:** It is the position of itself within an environment, this refers to estimating its position relative to a environment.

During the first steps of the research, I came across with an Article [3] where they present and explain a structure of the vehicle guidance system which it was used as a base of the architecture thought during this work, this architecture can it bee seen in Fig. 4.1

Even if in this article the focus was on fusion sensor and structure for the navigation and control of an autonomous vehicle, it gives a clear and concise structure of the vehicle and the fusion sensor, also giving some ground truth on the importance and benefits of using the data from both groups, Object-detecting sensors and the ego-position-detecting sensors, due

to the navigation concept proposed to use the data of the object-detecting sensors for the enhancement of the estimation of the vehicle ego-position.

Once the structure was defined the work and research focused on the object-detection system, as mentioned in Chapter 3 a series of algorithms and methods were considered in order to implement and obtain the desired methods, after it, as it is gonna be presented in the section 4.2 the method used didn't present the best results due to the way of extracting the ground element from the data, thanks to this a new objective came as to find a solution for the ground removal algorithm, as it is gonna be explained in the section 4.3.

4.1 Instrumentation

This project was developed with ROS Kinetic Kame in ubuntu 16.04 LTS, there were several libraries used but the main ones was the costmap_2d and the PCL libraries.

In order to do the development and the implementation of the algorithms, the data was obtained by means of the electric vehicle at which we had access. The car used for this project was the ZEDone from the brand ZHIDOU, which can be seen at the Fig.4.2

The sensors implemented and installed on the vehicle are:

- **Velodyne VLP 16:** The VLP-16 has a range of 100m, and the sensor's low power consumption (8W), light weight (830 grams), compact footprint (Ø103mm x 72mm), and a dual return capability. Velodyne's LiDAR Puck supports 16 channels, 300,000 points/sec, a 360° horizontal field of view and a 30° vertical field of view, with +/- 15° up and down.
- **ZED stereo camera:** The cameras has high-Resolution and High Frame-rate 3D Video Capture it also can have a depth Perception indoors and outdoors from 0.5m from the sensor at up to 20m. It is capable of spatial mapping and has a 6-DoF Positional Tracking.
- Continental radar ARS 408-21
- Leddartech laser sensors M16
- Texas instruments awr 1642
- STM HDR cameras

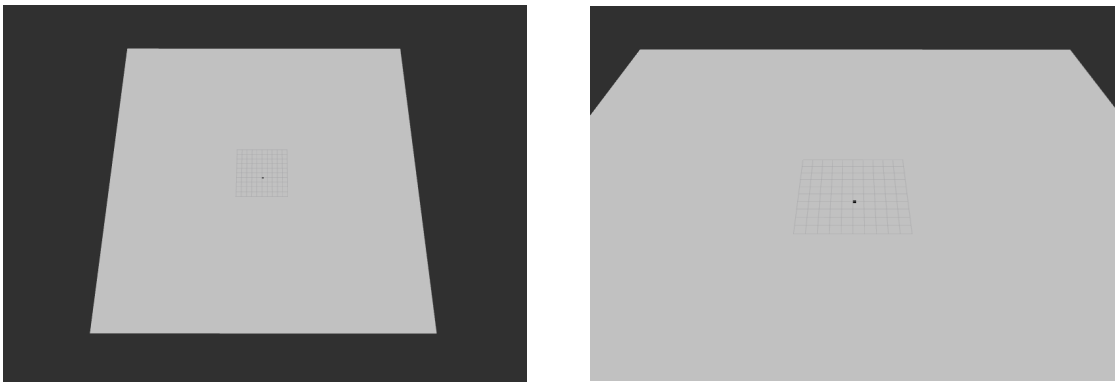
Figure 4.2 Electric vehicle ZED one

- STM GPS GNSS1a1
- swiftnav RTK gps

4.2 Occupancy Grid

The Occupancy grid is a map representation of the environment as an evenly spaced field of binary random variables each representing the presence of an obstacle at that location in the environment. In order to generate the grid map, an algorithm is needed to generate it from noisy and uncertain sensor measurement, with the assumption that the vehicle pose is known.

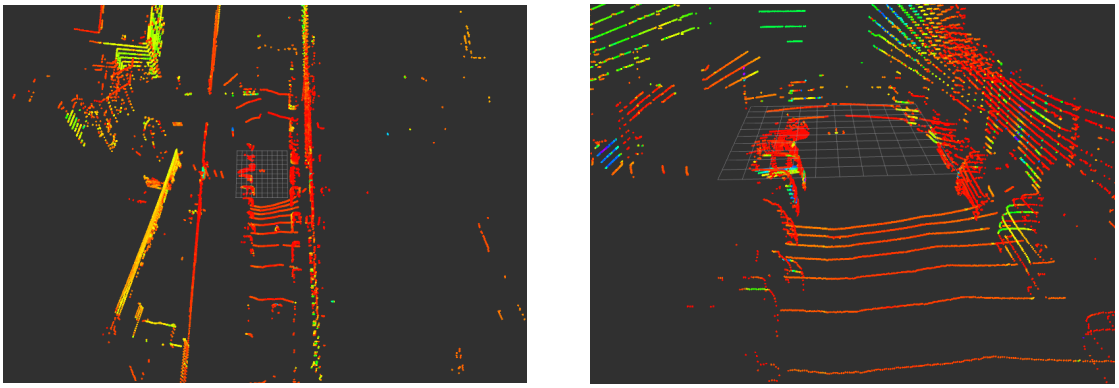
The first part needed was building an empty grid, the package used for visualizing in ROS, the occupancy grid and this project results was the rviz. With the empty grid and the data from the sensors the grid can be filled with the required data, for this project the grid was construct placing the vehicle in the center of the grid with a distance between it and the borders of the grid of 59.85 meters. An image of the empty grid can be see in the Fig.4.3,



(a) Far view of the empty grid for the Occupancy map.

(b) Far view of the empty grid for the Occupancy map.

Figure 4.3 Empty grid of the Occupancy map.



(a) Far view of the raw data.

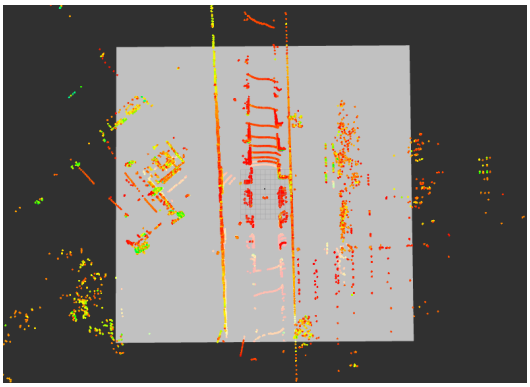
(b) Close view of the raw data.

Figure 4.4 View of the raw data.

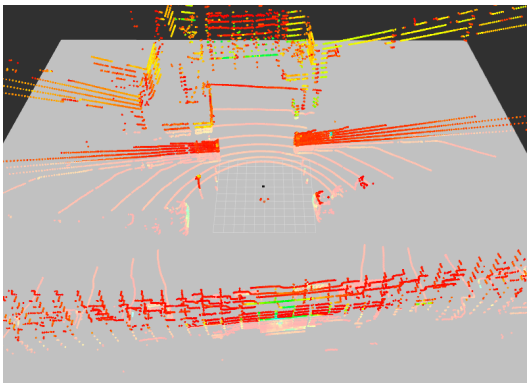
where it can also be seen a black point in the center of the map, point which represents the location of the sensor.

Following this procedure and once more time making use of the rviz package visualizer for ROS, the data was read and visualize, (see Fig.4.4) also centered and align with the empty occupation grid (see Fig.4.5), this alignment was done taking into account the center point of the empty grid and the actual position of the sensor.

Once the grid map and the data are align it can start identifying the objects present in the scene. It is important to have in mind what it was mention in chapter 3, section 3.1, about the characteristics and how the universal grid map works including the part about the precision. The higher precision is where it is constantly update with new measurements as is it in the

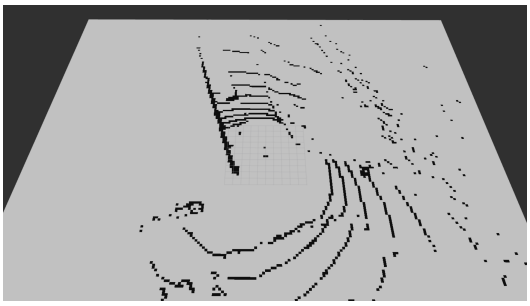


(a) Far view of the raw data align with the Occupancy grid.

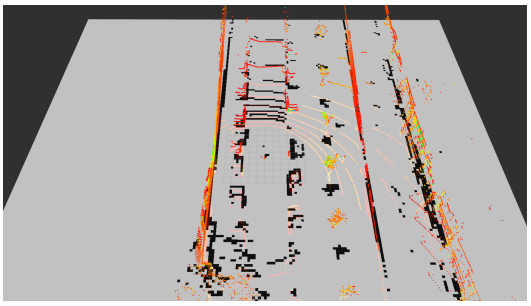


(b) Close view of the raw data align with the Occupancy grid.

Figure 4.5 View of the raw data align with the Occupancy map.



(a) The Occupancy grid recognizing the ground as an object.



(b) The Occupancy grid and input data recognizing the ground as an object.

Figure 4.6 Occupancy map with the ground as an object

region ahead of the sensor, for another hand the regions out of the sensor's field view present a decreased certainty due to drift of the robot's relative pose estimation.

As it can be seen in Fig. 4.6 the object detection of the system is working but with a big issue. The algorithms are detecting the ground points as an object, this ground points form the lines seen in the Fig. 4.6 where they surround the vehicle in a circular form as it is seen in figure 4.5b or in a more close view are the lines crossing the street in figure 4.4b.

Searching for a solution further of the proposed ground filtering, which is done by creating 2 pointclouds, the first one which contains the ground and the second that contains all the rest of the information. The ground filtering proposed it will be explained further in section 4.3.

4.3 Ground Removal

4.3.1 Segmentation

The first proposed algorithm for the ground removal was by making use of the SACSegmentation class present in the pcl_segmentation library in which, the SACSegmentation represents the Nodelet segmentation class for Sample Consensus methods and models, in the sense that it just creates a Nodelet wrapper for generic-purpose SAC-based segmentation.

Once the object of this class is created, the parameters for the segmentation object are set. First the type of model is selected as SACMODEL_PERPENDICULAR_PLANE, it determined a plane perpendicular to the desired axis, within a maximum specified angular deviation. The second parameter is the type of sample consensus method used for the segmentation which is the SAC_RANSAC. As RANSAC is an iterative method it was set the maximum iterations in 200 and a distance of 0.05 meters from the perpendicular plane of model and the hypothetical inliers. Finally the Y axis was set and a angular deviation of 0.4 radians of the specific plane (X-Z plane) coordinates that is perpendicular to the Y axis which is the ground plane.

After the segmentation object is created and set, the input pointcloud is filtered through the segmentation and thanks to the model it is obtained a set of data which corresponds to the ground, after it is obtained the original pointcloud get substracted by it producing a new pointcloud which contains all the data with the exception of the ground representation.

As it can be observed in the figure 4.7 the ground wasn't removed completely and still present false positive which can affect the proper functioning of the vehicle due to it considered as an object where in the reality there is non.

4.3.2 Normals

Since the method explained 4.3.1 presented the false positive issues, another algorithms was found, and based on the research, the implementation of a ground removal made by the analysis of the surface normals present in the environment give promising results.

For this implementation the pcl library was again used and the algorithm used started with the creation of the normal estimation class and passing the input data through it. The

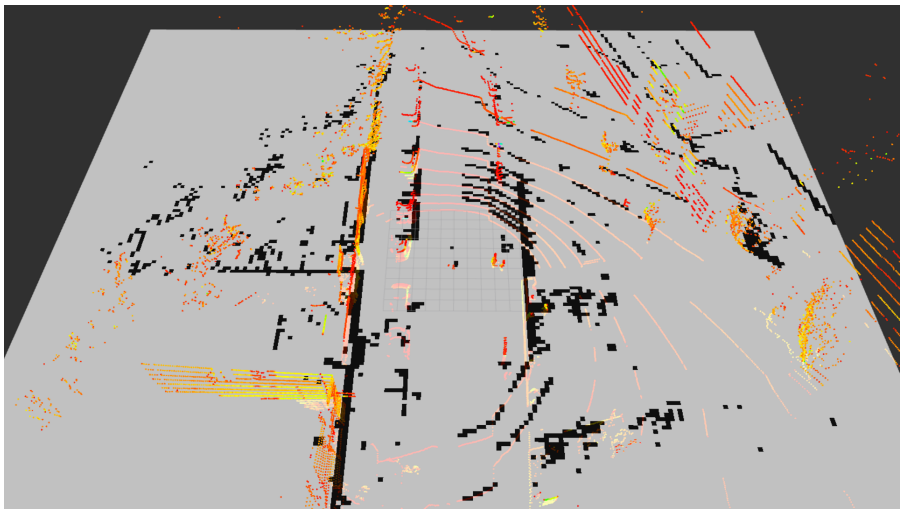


Figure 4.7 Occupancy grid with input data.

Normal Estimation as its name says, it estimates local surface properties (surface normals and curvatures) at each 3D point.

Once the input data was passed through the normal estimation an empty k-d tree representation is created in order to pass the normal estimation object with the purpose of filling the object based on the given dataset as no other search surface is given. For implementing the k-d tree algorithm it was used a sphere radius of 10 centimeters to find the neighbors from each of the points in the dataset and finally the new data is stored in a new pointcloud taking into account that the point size of the algorithm's result, the cloud of the normals have the same size as the point size of the original pointcloud.

With the cloud of normals the system eliminates the points which has the normal perpendicular to the X axis, removing in this way the ground, the normal estimation can it be seen in the figure 4.8 and figure 4.9.

4.3.3 Sum of Z-axis

It was considered that the cells in the grid which contains the data of the ground did not have a big amount of samples in its Z axis. With this in mind it was implemented one last algorithm in which it is store in a matrix the amount of point data that there is in each cell. With the matrix it is now possible to know which cell can be eliminated and which ones contain objects, making use of a low average and variance in height of cells to recognize the ground and with this generate the occupancy grid.

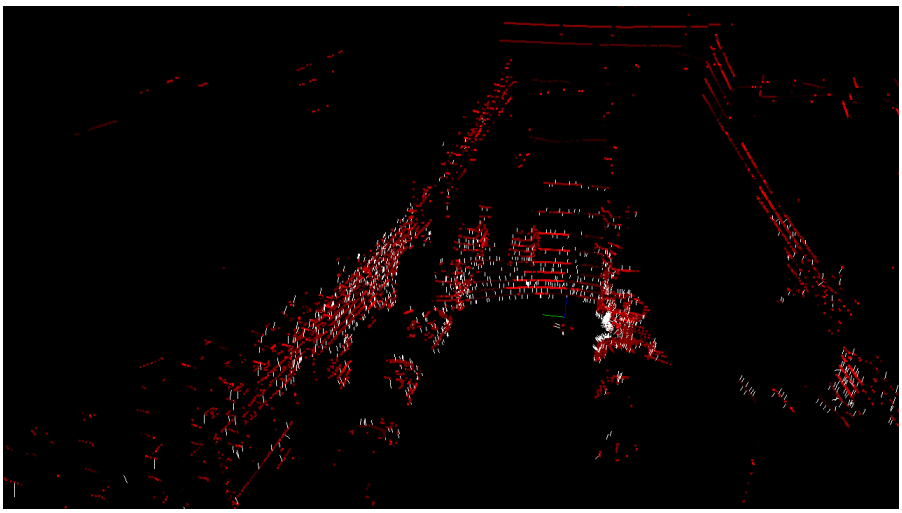


Figure 4.8 Far view of the original pointcloud with its normal estimation.

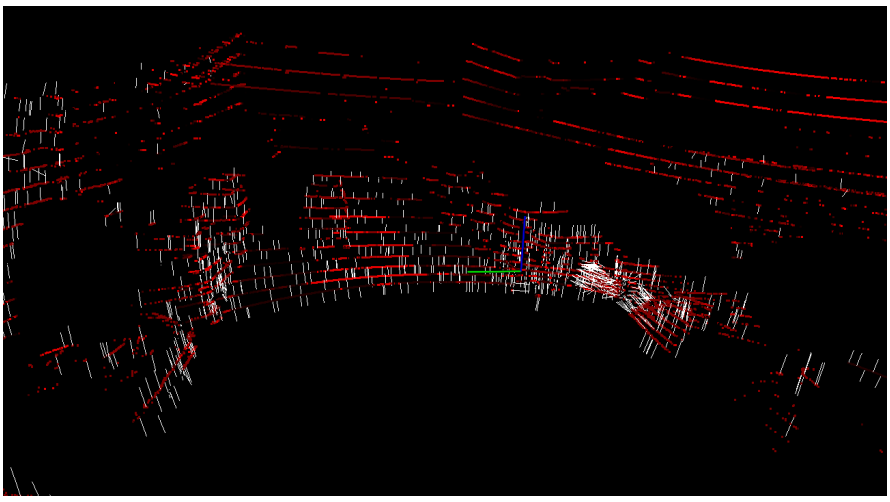
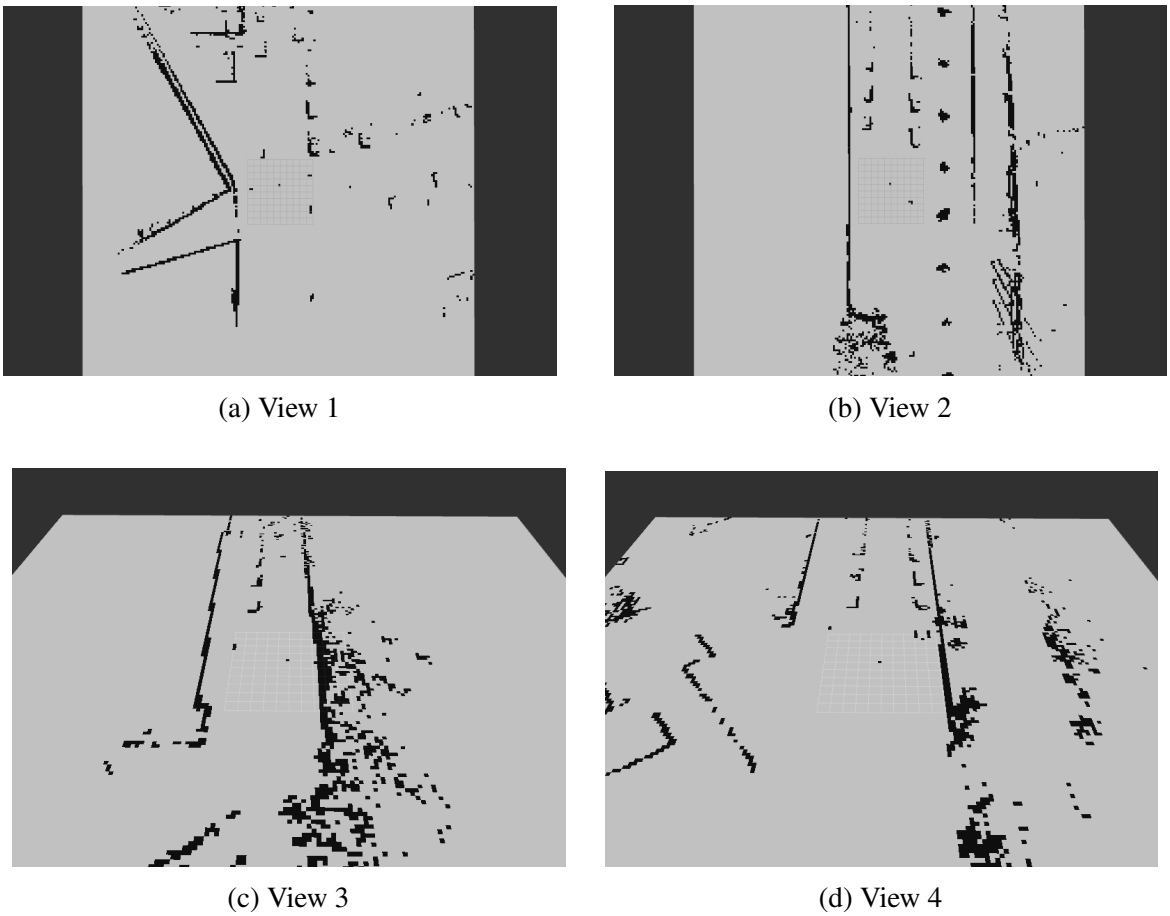


Figure 4.9 Close view of the original pointcloud with its normal estimation.



(a) View 1

(b) View 2

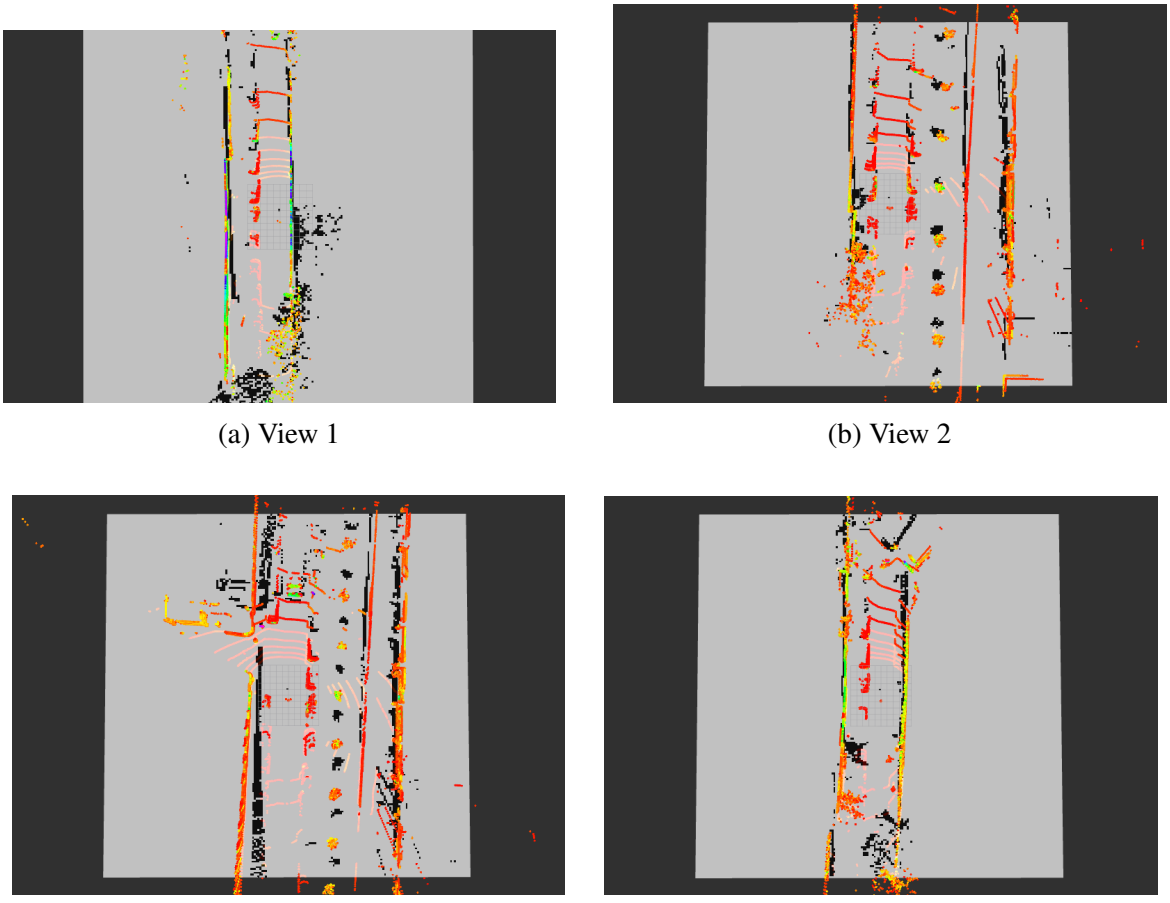
(c) View 3

(d) View 4

Figure 4.10 The Occupancy grid.

There were different approaches, a single threshold to divide the objects and non-objects, but it carried some issues because the decrease of point data due to the distance from the car was not considered. In order to correct this issue was proposed a fixed number of thresholds dividing the whole dimension of the occupancy grid, until finally it was implemented with a threshold decreasing with respect to the curve of the pointcloud's decrease data density.

The results of this algorithm can be seen in figure 4.10, figure 4.11 and figure 4.12.



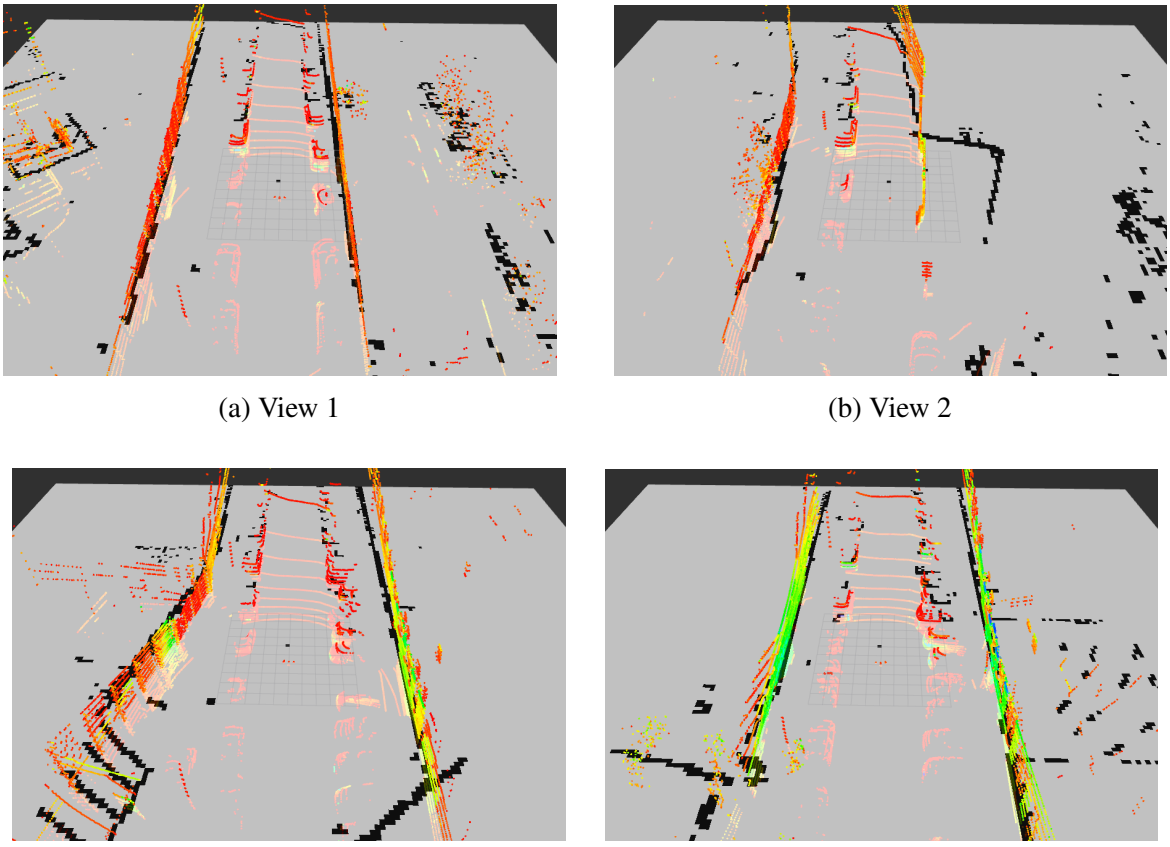
(a) View 1

(b) View 2

(c) View 3

(d) View 4

Figure 4.11 View of the raw data align with the Occupancy map.



(a) View 1

(b) View 2

(c) View 3

(d) View 4

Figure 4.12 Close view of the the Occupancy grid with the dataset.

Chapter 5

Conclusion and Future work

5.1 Conclusion

After starting the project with the idea of an autonomous ground vehicle and arriving to a narrow objective on Object detection, ground removal has led to realize the magnitude of a project like this.

Moreover, the project integrate different research groups and faculties of the university giving the opportunity to keep growing and get improved in future works and with the information gather here as the results of the implementation of the different algorithms it contributes in the improving and realization of the final objective which is the creation of an autonomous car.

This thesis has been an experience of learning and discovering this field of autonomous vehicles, its parts, structure and part of its implementation, from the point of the development was a very interesting work as I had to learn new topics, new platforms and systems.

In this thesis we proposed a system for occupancy grid creation based on those state of the art algorithms, we achieved those results by the implementation of the algorithms in open-source frameworks and the consideration of its improvement in the object detection and ground removal.

5.2 Future Work

As stated before, the work in an autonomous car has many opportunities and great potential, on each section of the vehicle can be improvements, starting with the data acquisition, with a more accurate and clean data in order to improve the whole pipe line and going all the way to the improvement of the design and structure of the whole vehicle without leaving on a side all the non-engineering contribution that it is done in this topic.

One activity can be improving the ground's removal of the data with a more robust algorithm minimizing the generation of false positives and true negatives, the approach with the estimation of the normals of each point is a path which can be expand and explode for this goal.

Bibliography

[1] Alireza Asvadi, L. Garrote, C. P. P. P. U. J. N. (2017). Multimodal vehicle detection: fusing 3d-lidar and color camera data l_1. *Pattern recognition letters*, 000:1–10.

[2] Alireza Asvadi, Paulo Peixoto, U. N. (2015). Detection and tracking of moving objects using 2.5d motion grids l_1. *IEEE 18th International Conference on Intelligent Transportation Systems*, pages 788–793.

[3] Andreas Simon, Ina Sohnitz, J. C. B. W. S. (2000). Navigation and control of an autonomous vehicle l_1. *IFAC Control in Transportation Systems*.

[4] Andrei Furdaand, L. V. (2010). An object-oriented design of a world model for autonomous city vehicles l_1. *IEEE Intelligent Vehicles Symposium*.

[5] Armin Hornung, Kai M. Wurm, M. B. C. S. W. B. (2013). Octomap: an efficient probabilistic 3d mapping framework basedon octrees l_1. *Autonomous Robots*.

[6] Charles R. Qi, Li Yi, H. S. L. J. G. (2017a). Pointnet++: Deep hierarchical feature learning on point sets in a metric space l_1. *arXiv:1706.02413*.

[7] Charles R. Qi, Hao Su, K. M. L. J. G. (2017b). Pointnet: Deep learning on point sets for 3d classification and segmentation l_1. *arXiv:1612.00593*.

[8] Elfes, A. (2013). Occupancy grids: A stochastic spatial representation for active robot perception l_1. *arXiv:1304.1098*.

[9] Hernan Badino, Uwe Franke, D. P. (2009). The stixel world - a compact medium level representation of the 3d-world l_1. *31st DAGM Symposium on Pattern Recognition*.

[10] International, S. (2018). Taxonomy and definitions for terms related to driving automation systems for on-road motor vehicles. l_1. $J3016_2$01806.

[11] Kaustubh Pathak, Andreas Birk, J. P. S. S. (2007). 3d forward sensor modeling and application to occupancy grid based sensor fusion l_1. *IEEE International Conference on Intelligent Robots and Systems*, pages 2059–2064.

[12] Marius Cordts, Timo Rehfeld, L. S. D. P. M. E. S. R. M. P. U. F. (2017). The stixel world: A medium-level representation of traffic scenes l_1. *arXiv:1704.00280*.

[13] Mohammad Haghighat, Mohamed Abdel-Mottaleb, W. A. (2016). Discriminant correlation analysis: Real-timefeature level fusion for multimodal biometric recognition l_1. *IEEE TRANSACTIONS ON INFORMATION FORENSICS AND SECURITY,*, 11:1984–1996.

[14] Péter Fankhauser, M. H. (2016). A universal grid map library: Implementation and use case for rough terrain navigation l_1. *The ROS Multimaster Extension for Simplified Deployment of Multi-Robot Systems*.

[15] Qadeer Baig, Mathias Perrollaz, C. L. (2014). A robust motion detection technique for dynamic environment monitoring: A framework for grid-based monitoring of the dynamic environment l_1. *IEEE Robotics Automation Magazine*, 21:40–48.

[16] Radu Bogdan Rusu, S. C. (2011). 3D is here: Point Cloud Library (PCL). In *IEEE International Conference on Robotics and Automation (ICRA)*, Shanghai, China.

[17] Uwe Franke, David Pfieffer, C. R. C. K. M. E. F. S. R. G. H. (2013). Making bertha see l_1. *International Conference on Computer Vision*, pages 2014–221.

[18] Yin Zhou, O. T. (2013). Voxelnet: End-to-end learning for point cloud based 3d object detection l_1. *arXiv:1711.06396*.