COLLISION AVOIDANCE FOR AUTOMATED VEHICLES USING OCCUPANCY GRID MAP AND BELIEF THEORY

by

Reza Soltani

A Thesis

Submitted to the Faculty of Purdue University

In Partial Fulfillment of the Requirements for the degree of

Master of Science

Department of Electrical and Computer Engineering

Indianapolis, Indiana

August 2021

THE PURDUE UNIVERSITY GRADUATE SCHOOL
STATEMENT OF COMMITTEE APPROVAL

Dr. Lingxi Li, Chair

Department of Electrical and Computer Engineering

Dr. Sarah Koskie

Department of Electrical and Computer Engineering

Dr. Yaobin Chen

Department of Electical and Computer Engineering

Approved by:

Dr. Brian King

This work is dedicated to my dear wife and
my two sweet sons Sepehr and Sina.

ACKNOWLEDGMENTS

I also, would like to dedicate this work to my dear parents Mr. Karam Ali and late Mrs. Sarah Shokri, my siblings especially my senior brother Fazlollah and his wife Havva, and my in-laws Mr. Asghar Kosari and Mrs. Tahereh Asadi, who believed in my dreams, encouraged me, and took my hand to raise up and move on forward. Special thanks to my dear wife Maryam with her unconditionally unwavering support, without her help doing this job was impossible. Thanks to my great sons for enduring me spending less time with them due to being school. Tanks to my other family members that took my hand through this journey.

Special thanks to Dr. Lingxi Li, my dear professor and thesis advisor, with his knowledge and patience has thought me how to control and manage myself to solve problems. Without his support, this job was not possible to be done. Special thanks to our great department chair Dr. Brian King, my program adviser who designed a great algorithm for running the department with his extraordinary knowledge and management. He compassionately and honestly does his best to advance the department. Special thanks to dear Dr. Yaobin Chen, for his hard working and being one of the committee member. I really appreciate for letting me be in your wonderful and useful class. My special thanks to Dr. Sarah Koskie for her diligent effort in educating the students. It is a great honor for me that you are a member of the committee. Special thanks to my professor and friend Dr. Chris Sexton for his great support. Thanks to all my professors for their effort to make the society better under any circumstances. At the end, my special thanks and gratitude to dear Mrs. Sherrie Tucker for her extraordinary support, following-up, patience and humbleness. She is a valuable treasure for the ECE department. Undoubtedly, the department owes much to her effort.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF SYMBOLS

ω	Angular velocity
h	Height
m	Mass
m_{i}	Occupancy grid cell
θ	Pitch angle
ρ	Roll angle
v	Velocity

ABBREVIATIONS

ATTO	Advance time-to-Occupancy
AV	Autonomous Vehicle
CAS	Collision Avoidance System
DST	Dempster-Shafer Theory
DAS	Driving Assistant System
FOD	Frame of Discernment
GPS	Global Positioning System
IMU	Inertial Measurement Unit
IGM	Investigated Grid Map
OGM	Occupancy Grid Map
POGM	Predictive Occupancy Grid Map
ROS	Robot Operating System
SAE	Society of Automotive Engineers
TTO	Time-to-Occupancy

ABSTRACT

Decades ago, talking about the vehicle that can drive without human being intervention or self-driving cars seemed funny, unrealistic and unlikely compare to the airplanes because of the obstacles. Nowadays, researchers, have made it a reality by utilizing of sciences such as computer, electronic, mechanic etc. despite of massive progress, it seems that we are still at the beginning of the road. Huge number of accidents that occur due to human errors is a proof of its correctness. Lack of sufficient preventative driving aid such as collision avoidance system is one of the significant reason. Most of the collision happens when human drivers make mistakes. On the other hand, autonomous vehicles (AVs) will play an important role in future travels, and more accident preventatives are needed.

For driving assistant system (DAS), an ego vehicle extremely needs a detailed environment representation especially in crowded urban area for static and dynamic object detection and free space determination. Due to inaccurate pose estimation and noisy measurements by using Lidar sensor scanning, some uncertainty is generated because of the perception of the new areas and errors. When a lidar generate multi-layer data and eco it is possible to get complexity increase. Different sources of uncertainty need to be managed. otherwise, collision is happening. For that reason, a beneficial representation of drivable area is offered to navigate autonomous vehicles. That would be occupancy grid map (OGM) which plays major role for the obstacle avoidance.

A collision avoidance system (CAS) is the best way to estimate the risks associated with the surrounding obstacles and guides the AV into a safer path when the ego car encounter with a dangerous situation. CAS provides safe region information and consequently collision-free area in the presence of other obstacles such as cars, pedestrian, signs etc. There are some common CAS functions forward and rear collision warning and emergency braking that can help to prevent some major and minor accident. These techniques have been developed

lately. However, they are not sufficient and reliable preventative for limiting or avoiding unexpected potential crashes with the surrounding obstacles.

This thesis discusses OGM, CAS and belief theory, and propose some of the latest and the most effective method such as predictive occupancy grid map (POGM), risk evaluation model and OGM role in the belief function theory with the approach of decision uncertainty according to the environment perception with the degree of belief in the driving command acceptability. Finally, how the proposed models mitigate or prevent the occurrence of the collision.

1. INTRODUCTION

After the industrial revolution, human being took steps to achieve their dreams and possibilities. Human's endeavors to achieve the new technologies are yielding result, and new windows are being opened to discover the unknowns every day. Undoubtedly, the automotive industry, as an attractive beneficial subject, has tempted many researchers to redouble their efforts. Today, the different types of self-driving cars tell us that researchers have made significant progress.

In the United State, as one of the leaders of automotive industry, Autonomous Vehicle (AV) technology and intelligent transportation system rapidly is growing up and they are parading on the roads. Many states such as California, Arizona, Nevada, and Florida etc. host these cars to pass their test drive. AV manufacturers are working hard to update the autonomy levels to change their semi-autonomous car to the higher levels (Favarò et al., 2016) [1].

1.1 Thesis contributions

Collision avoidance without existing a reliable occupancy grid map is impossible. Development and improvement of the grid map is the main goal of researchers. The primary contributions of this thesis are summarized as follows:

- Some important challenges in machine learning algorithm in AV industry have been identified and some solutions are proposed in order to limit them.
- Estimation of occupancy probability for each cell by applying binary Bayes rule is developed theoretically.
- 2D occupancy grid map with the laser scan algorithm is developed and probability of each cell is calculated in 0° to 360° in the MATLAB.
- 3D occupancy grid map algorithm with inflating the map is developed.

- The algorithm of inflation of coordinates (obstacles) in both binary and normal occupancy grid is developed in the MATLAB.
- Generating a grid map by stereo vision algorithm and depth calculation by epipolar geometry are discussed in detail.
- An obstacle tracking model is developed for dynamic object. The Kalman filter theoretically is developed and discussed in detail, and it's algorithm is developed and implemented in the MATLAB to track the object.
- Dempster-Shefer Theory in combination of mass functions is improved and belief function improvement for modifying basic probability assignment is discussed in detail with an example.

1.2 Chapter organization

In four chapters of this thesis, these topics are discussed as summarized below: Chapter one summarizes the phased growth of AV and the information acquisition tools from the surrounding space. Second chapter describes how to generate an occupancy grid map with a probabilistic approach and computer vision system. In chapter three some major tools to prevent or mitigate a collision is discussed and finally in chapter four generating a belief map and its application to prevent a collision.

1.3 Automation levels

According to the Society of Automotive Engineers (SAE) definition, a car considers as an autonomous vehicle, which be qualified to stay on the six levels of autonomy scope where the autonomous technology be able to support and perform the assigned driving duties. They are adopting a technology that is capable to assist a human driver in duties such as:

1. Controlling the speed and steering of the car.

2. Be able to monitor and gain the surrounding information (e.g., all kinds of obstacles such as vehicles, buildings, pedestrian, and road signs, etc.).

These tasks are related to each other (e.g., the dependency of acceleration/deceleration with the traffic light). The summary of six levels of autonomy that are accepted by governments and car makers is stated in the below Figure [2].

SAE level	Name	Narrative Definition	Execution of Steering and Acceleration/ Deceleration	*Monitoring* of Driving Environment	Fallback Performance of *Dynamic Driving Task*	System Capability *(Driving Modes)*
***Human driver* monitors the driving environment**						
0	**No Automation**	the full-time performance by the *human driver* of all aspects of the *dynamic driving task*, even when enhanced by warning or intervention systems	Human driver	Human driver	Human driver	n/a
1	**Driver Assistance**	the *driving mode*-specific execution by a driver assistance system of either steering or acceleration/deceleration using information about the driving environment and with the expectation that the *human driver* perform all remaining aspects of the *dynamic driving task*	Human driver and system	Human driver	Human driver	Some driving modes
2	**Partial Automation**	the *driving mode*-specific execution by one or more driver assistance systems of both steering and acceleration/ deceleration using information about the driving environment and with the expectation that the *human driver* perform all remaining aspects of the *dynamic driving task*	**System**	Human driver	Human driver	Some driving modes
***Automated driving system* ("system") monitors the driving environment**						
3	**Conditional Automation**	the *driving mode*-specific performance by an *automated driving system* of all aspects of the dynamic driving task with the expectation that the *human driver* will respond appropriately to a *request to intervene*	System	**System**	Human driver	Some driving modes
4	**High Automation**	the *driving mode*-specific performance by an automated driving system of all aspects of the *dynamic driving task*, even if a *human driver* does not respond appropriately to a *request to intervene*	System	System	**System**	Some driving modes
5	**Full Automation**	the full-time performance by an *automated driving system* of all aspects of the *dynamic driving task* under all roadway and environmental conditions that can be managed by a *human driver*	System	System	System	**All driving modes**

Figure 1.1. Six different levels of autonomy in autonomous vehicle [2].

In the autonomy levels which start from level 0 and goes to the level 5, each level includes the specific characteristics, different operations and driving assistant technology. There is no automation in level 0 and level 5 is fully automated [2].

- Level 0: There is No Driving Automation. It is controlled manually, and the dynamic driving task is done by human being. Some system such as emergency break helps the driver.
- Level 1: The vehicle is featured with the lowest level of automation. Driver assistance such as cruise control and adaptive cruise control that helps the driver to keep the car at a certain distance with the front car is designed at this level.
- Level 2: Both steering and acceleration can be controlled by the car, that means advanced driving assistance system (ADAS) is designed at this level.
- Level 3: Turning from ADAS to conditional driving automation was a big jump that happened at this level. The vehicles are capable to detect the environment.
- Level 4: The car is equipped with high driving automation system such that driver interaction is not required in most case. Automation system tasks can be override by driver.
- Level 5: Human attention and dynamic driving responsibility is canceled at this level and full driving automation feature is provided at this level.

1.4 Automation vehicle market

It is not an exaggeration to say that AV technology's market is one of the fastest growing market among other technologies because of improvement of several substantial transportation problems such as:

- Mobility: Having option in transportation with a good quality in option such as time and affordability is stabilized here.
- Safety: Improving safety of the roads is one of the good outcomes of autonomous technology market. Big percentage of accident caused by human error is because of road safety issue.

- Efficiency: Reducing congestion on roads, resulting several good outcomes such spending less time in traffic, consuming less fuel, reducing cost of road and etc.
- Enhancement of traveling quality: Entire or part of the driving time could be dedicated to other duties [2].

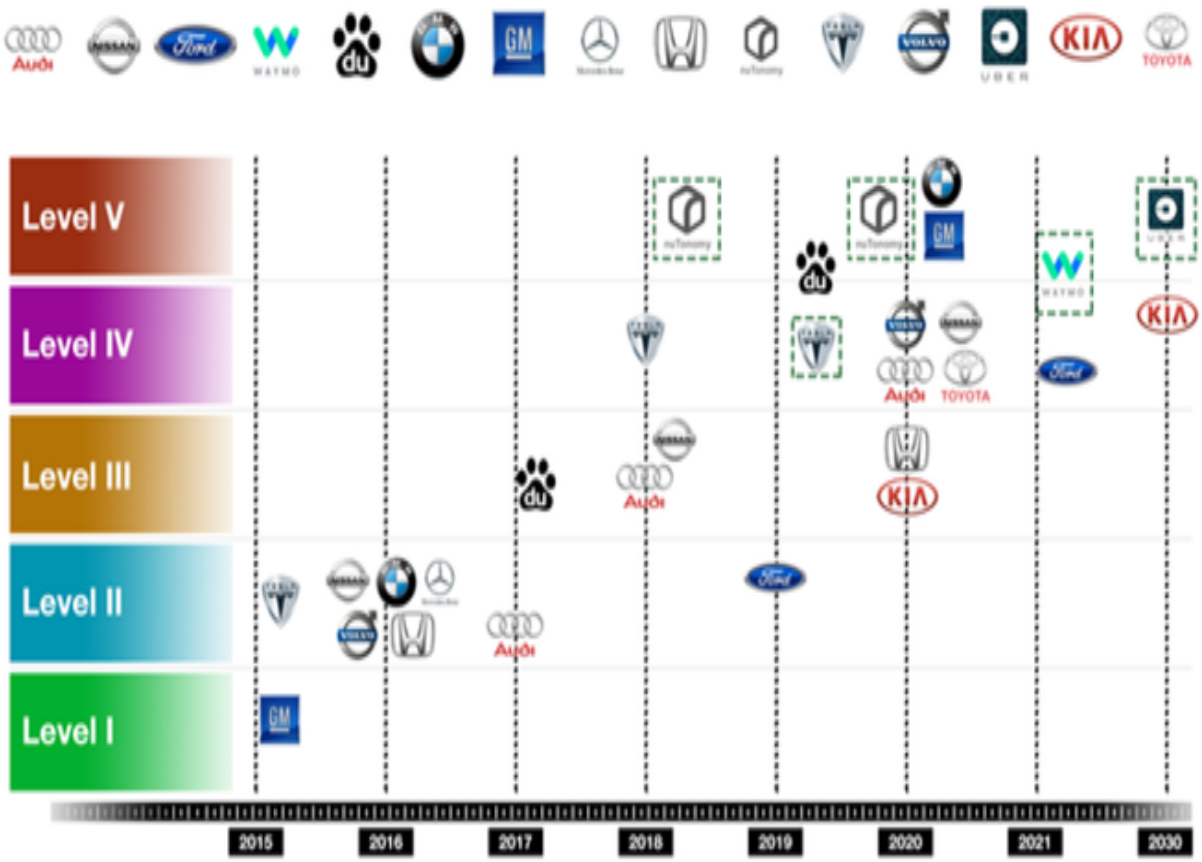

Figure 1.2. Overview of AV market, 2015–2030 estimated timeline [2].

1.5 Algorithm in AV

Nowadays, machine learning algorithms are extensively used to solve most challenging problem and in AV industry it is one of the most widely used. With the help of sensor data processing center, it is possible to massively raise the application of machine learning and do more and new task.

In autonomous driving, algorithms have made significant progress in three major areas: Sensing, perception, and decision. Sensors' job is to collect unprocessed and raw information, in sensing process, the important data and information is extracted for next stage use. Perception is perceiving the environment and its interaction with AV. In decision, driving task is done to make appropriate plan for reaching to destination safely [3].

1.5.1 Sensing

Without a doubt, an autonomous car without sensor is just an ordinary car. In an autonomous vehicle, several sensors are interacting to each other till AV does its job based on its design purpose [3]. Each sensor does specific operation. The collected information from different sensor must be incorporated to get authentic information for security.

GPS and IMU

Lack of accuracy in the navigation system can be harmful in driving. So, with Global Positioning System (GPS), the car will be localized, and its position is determined with the geographical coordinate's assistance. There is some problem with the GPS signal that can be affected by environment change which causes the signal to be lost or become week. To solve the GPS signal lost issue, Inertial Measurement Unit (IMU) is used which consist of six axis, acceleration and Gyroscopic axis in x, y and z direction. It provides yaw rate, pitch rate and roll rate. IMU provide the signal while GPS has no signal. They restrict the AV by detailing. The accurate localization is given by both GPS and IMU. GPS has slow update rate and IMU has higher update rate [3],[1].

Lidar

Light detection and ranging (Lidar) measures the infrared light reflection off of objects and scans its surrounding environment in 360° angle. It creates multiple beams that emits at angles together. Rotating sensor spins inside of a compartment to generate a 3D picture of its surroundings. Because of the shorter wavelengths, Lidar is the most accurate sensor. Mapping, obstacle avoidance and Localization are the duties that assigned to Lidar. The operation principle of Lidar is, to calculate the returning time of beam from obstacle to the

emission source. Because of high accuracy, HD maps are generated. It helps to localize a moving car and detecting the obstacle in front of it [3].

Cameras

Object detection and its tracking is one of the most essential, complicated, and challenging in the autonomous driving system. This significant duty is assigned to camera. They provide multitude of data about the surrounding environment. Obstacle detection, sign detection, pedestrian detection etc. are the jobs that camera does that [3]. AV are designed to drive better than human driver so they must see better than human driver to do it. AV manufactures are updating car security with installing multiple cameras around the car with resolution higher than 1k $\times$1k to diagnose all objects and signs in 360° around the cars. There are some disadvantages of camera such as having problem in detecting non-illuminated condition.

Radar and Sonar

Radar emits electromagnetic waves, and **sonar** sends out acoustic waves. In radar and sonar, reflected radio waves is used to sense surrounding objects [3]. They are generally utilized for short range object detection. Because of the permeability of the waves, radar is more useful than the other sensing tools in bad weather conditions. Radar sensors detect metal objects better than other sensors. Radar enables the cars to detect long-range that can detect objects through dusty, foggy, rainy, and snowy weather. The generated data and information by radar and sonar does not need heavy processes. Radar and sonar are used in the breaking applications.

Sensor	Accuracy	Environment Susptibility	Cost	Reliability	Dynamic Range
▪ GPS	-	--	+++	+++	+++
▪ Radar	++	--	++	++	+
▪ Camera	--	--	+	+	+
▪ ultrasonic	-	--	+++	++	--
• Lidar	++	-	---	-	-
• IMU	+	++	+	+++	+++

Figure 1.3. Comparison of sensors competency [3].

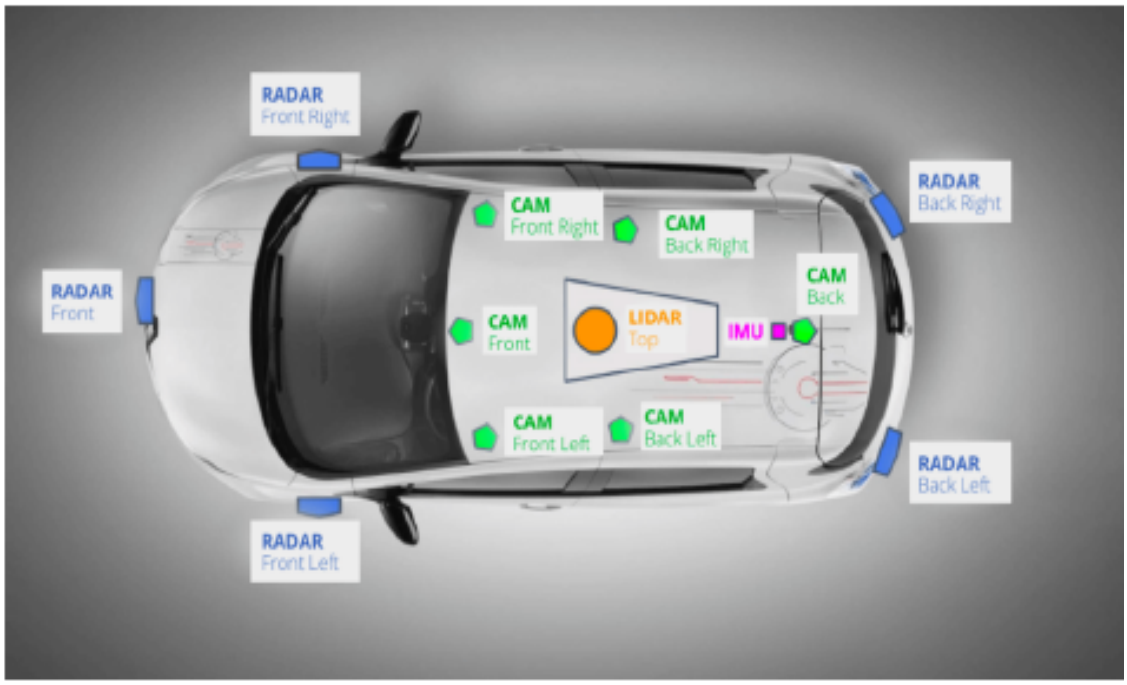

Figure 1.4. Sensors location in AV [4].

Despite the fact that the ego cars are designed based on the most accurate researches and equipped with the latest technologies, but still they are facing many challenges and restrictions such as:

- Because machine learning algorithms are being developed day by day, therefore, the potential scenario can not be predicted.
- Because the criterion of superiority of the algorithms is the optimality of their time complexity, therefore, different software-defined AV's might have different amount of computational performance at the same time.
- Permanent structural and environmental constraints for sensors such as software issue or noisy and foggy environment.
- Lack of full cooperation of AV manufacturers in the development of required network and infrastructures causes serious risks always.
- The necessary training for taking back the control in case of need to maintain safety is not provided by manufactures in advance.

However, overcoming these limitations are impossible due to their nature and potential, but they can be reduced in term of accurate and dynamic management. Some helpful suggestions are provided below:

- Using a well designed, uniform or compatible machine learning algorithm and AI system if possible.
- Applying a sophisticated and reliable hardware to lower the computational and time complexity down.
- Providing software update remotely.
- Providing regular, inexpensive and reliable after-sales services.

2. OCCUPANCY GRID MAP

One of the most significant goals of car industry researchers, in addition to make driving easier, is to mitigate or eliminate car collision which causes a lot of life losses and financial losses to people every year. Increasing safety has been one of the main motivations of car-makers in recent years and has led this industry to a fastest-growing research area. Although human error one of the leading causes of car collisions but lack of some tool such as occupancy grid map (OGP) has greatly increased its number.

Nowadays, sensors are one of the most important methods of obtaining information among the existing methods from the environment. Two groups of sensors are considered: laser sensor and sensor based on computer vision. Laser sensor provides measurement with the high accuracy, but the information obtained through the laser sensors is not sufficient to classify surrounding objects correctly. On the other hand, there is a system that has a lot of rich information that the sensors in this system are based on computer vision. One of the most efficient method for detecting the obstacles around the AV is stereo vision algorithms. Free area ahead the car is detected by this method too. According to the most authors in the AV industry, the information obtained by computer can be represent as the free space map and the obstacle map [5].

2.1 Structure

As it is mentioned earlier, machine learning algorithm (algorithm) is the foundation of problem solving in AV industry and a lot of other sciences. Occupancy grid mapping (OGM) in fact is family of computer algorithms in probabilistic robotics that applies in AV and mobile robots. These algorithms determine the problem of generating maps from uncertain and noisy sensors measurement information. In this case, it is assumed the vehicle's pose is already known.

2.2 Basic Idea

In a moving AV on the drivable area, the data that is collected from the ego car is considered for further research. An autonomous car is equipped with all kinds of on-board sensors that play the major role on data collection. Lidar, as a one of the most important range sensors is mounted at the top of the autonomous car that emit some laser rays in some pre-determined directions and receive their reflection to obtain the traveled distances. Rays' travel is depending on free region in the direction of the rays. It traverses longer distances if objects are far away in their direction and travel shorter distances if there are any obstacles in its direction. As the autonomous car collects this data and information over the time while moving around, it generate the objects when the rays are blocked by the objects. Following sections, explains how the OGM is build up with the laser reading.

2.3 Map Size

Based on the application, different size of maps with the different resolution is generated. In terms of dimensions, there is two types of maps [6].

- 3-dimensional occupancy grid map (3D OGM)
- 2-dimensional occupancy grid map (2D OGM)

2D OGM is represented by two categories.

- Binary occupancy grid
- Probability occupancy grid

With binary occupancy grid, a 2D occupancy map object is generated. It can be used to represent and visualize an ego car driving area and the obstacles around it. Path planning and mapping applications are the algorithms where the occupancy grids are used. Mapping application contains finding collision-free paths, collision avoidance performance and localization calculation. In the localization, world map is given, and it is asked to find ego car

position in it. Conversely, in grid mapping, ego vehicle position is given through the sensors and it is asked to find the world map [7].

2.4 Random variable

before moving to next topic, the definition of random variable with a simple example is reminded.

Definition: In probability, random variable (RV) is a function that maps sample space members to the real numbers.

Example:

In tossing a coin, there are two outcomes or states: Head or Tail. The probability of each state could be any amount between 0 and 1. If Head is abbreviated with H and Tail is abbreviated with T, the sample space here will be S= {H, T}. Instead of representing with H and T, let us assign a numeric value to the H and T arbitrarily:
Suppose H→0 and T→1 if we do this mapping with a function like X, we can have.
X: {H, T}→{0, 1} (X is a function that maps sample space to real numbers).
X= {0, 1} is called random variable. Instead of 0 and 1 we can use any arbitrary values.

$$X = \begin{cases} 0 & \text{if } H \\ 1 & \text{if } T \end{cases}$$

2.5 Occupancy in terms of probability

Occupancy is defined as a binary random variable. Binary random variable means the only number is used here are 0 and 1. In this case, occupancy defines a probability space that has two possible states: Free and Occupied. The occupancy random variable has two values 0 and 1 [8]. An occupancy grid map is an array of occupancy variables like a computer memory structure. Each element of grid can be represented by the corresponding occupancy variable. If each member or cell is indicated by m_{i}.

- Occupancy: binary RV

m_i : {free, occupied}→{0,1}

Below figure shows 2D occupancy grid map:

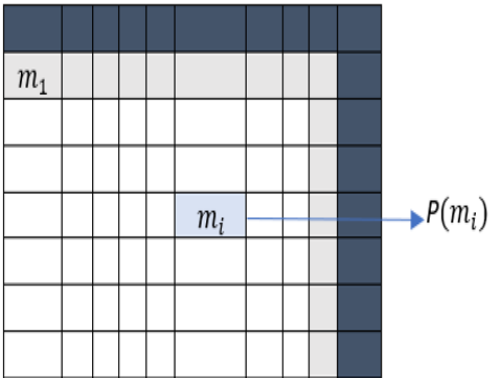

Figure 2.1. Occupancy grid map.

Occupancy grid map is discrete cells that surrounds AV position, and it is a main prerequisite to do motion planning task. If the static objects around the AV, such as trees, building and road signs are existing in the cell in grid location it is classified as occupied cell. Each cell maps to a binary value which 1 tells that square is occupied by a static object and 0 indicates that is free. To create a precise occupancy grid mapping, it is supposed to all moving objects is deleted from sensor data and occupancy situation of one cell doesn't effects other cell. See Fig. 2.2 [9].

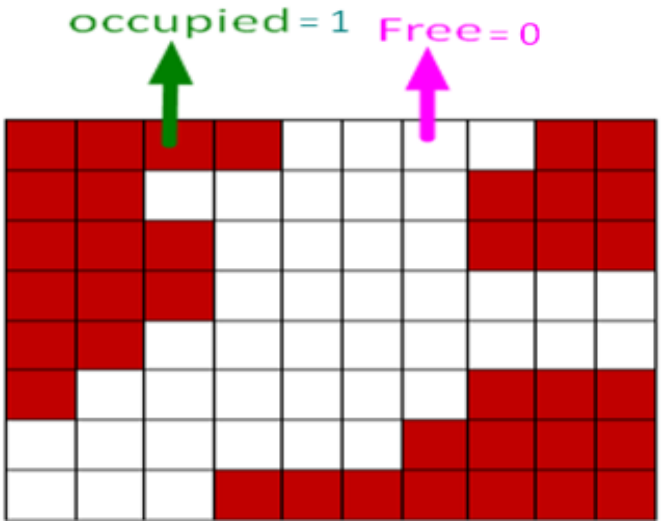

Figure 2.2. Occupancy situation of a cell .

$m_{\mathrm{i}} \in \{0,1\}$

Although the occupancy grid map is one of the good tools for collision avoidance but there are some restrictions such as: There is no state for partially empty cell or partially occupied cell [9]. Sometime the emitted rays by sensors are not reflected by some transparent or partially transparent obstacles such as glass or some obstacles, do not allow laser light to be reflected due to their appearance. One of the simple solution is adding the measurement time to the continuously measure the laser rays with the low intensity. With counting the number of reflections 1 and passes 0, the probability of occupancy is gained which is number of reflections divide by the sum of number of reflections and passes (entire emitted laser beams). With this approach, dedication an area with high probability or using fractional numbers for each state (occupied or free) are recommended in order to solve the problem.

2.6 Estimation of occupancy probability of a cell

Distance information between an AV and drivable environment or cells are calculated by OGM algorithm and sensor, based on ray passage through the cells. If a measure is defined for distance, it would be 0 when ray pass through the cell and 1 when it hits the cell. There

are 4 types of conditional probability between cell m_i and measurement γ because both have binary values 0 and 1 (2×2=4).

γ={0,1} and m_i={0,1}

P(γ=1 | m_i=1) which is probability of γ=1 given m_i=1 means probability of an occupied measurement is 1 when cell is occupied by an obstacle (ray hit to the occupied cell).

P(γ=0 | m_i=1): probability of free measurement for an occupied cell (ray doesn't hit the occupied cell).

When m_i is occupied means it is 1 and there may be a measurement for that, means γ is 1 too. So, it can be said it is true occupied. Same reason for other probabilities. In general:

P(γ=1 | m_i=1) : is true, occupied measurement
P(γ=0 | m_i=1) : is false, free measurement
P(γ=1 | m_i=0) : is false, occupied measurement
P(γ=0 | m_i=0) : is true, free measurement

As it is mentioned, in static state, each cell in grid map is a binary random variable and it models the occupancy.

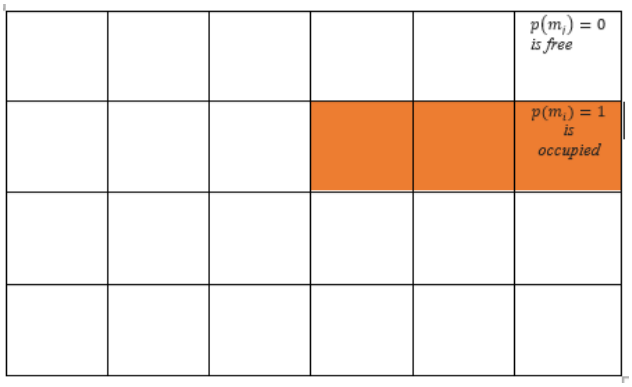

Figure 2.3. Occupancy position of a cell.

If $p(m_i) = 0.5$ it is said no information about it.

The probability distribution of the map is calculated by the product of probability of each cell [10].

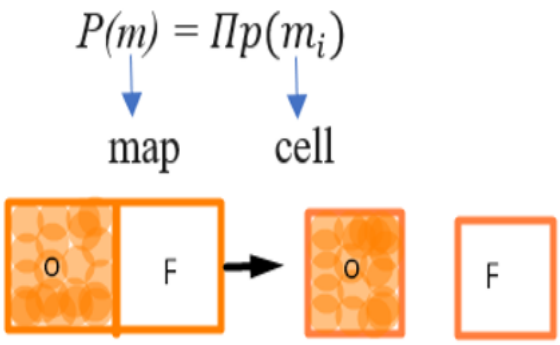

Figure 2.4. Probability of a map.

By using binary Bayes filter and measurement it is possible to estimate occupancy probability for each cell. before that, it is needed to calculate the posterior probability which is:

$p(m_{\mathrm{i}} \mid \gamma_{1:t}, x_{1:t})$

where m_{i} is the map and $\gamma_{1:t}$ is set of measurements from time 1 to t and $x_{1:t}$ is the set of autonomous vehicle poses from time 1 to t (information must be given).

$p(m_{\mathrm{i}} \mid \gamma_{1:t}, x_{1:t})$ is the posterior and the goal is to calculated it.

The posterior of entire map is approximately calculated by:

$$p(m \mid \gamma_{1:t}, x_{1:t}) = \Pi p(m_{\mathrm{i}} \mid \gamma_{1:t}, x_{1:t}) \qquad (2.1)$$

The occupancy of a grid cell is calculated based on given measurement below [10].

$$p(m_{\mathrm{i}} \mid \gamma) = \frac{p(\gamma \mid m_{\mathrm{i}})p(m_{\mathrm{i}})}{p(\gamma)} \qquad (2.2)$$

represent the occupancy probability of each cell from the given measurement.

where :

$\mathrm{p}(m_{\mathrm{i}})$ is prior map

$\mathrm{p}(m_{\mathrm{i}} \mid \gamma)$ is posterior map

$\mathrm{p}(\gamma \,|\, m_\mathrm{i})$ is likelihood

$\mathrm{p}(\gamma)$ is evidence

Bayes rule:

$$p(m_\mathrm{i} \,|\, \gamma_{1:t}, x_{1:t}) = \frac{p(\gamma_t \,|\, m_\mathrm{i}, \gamma_{1:t-1}, x_{1:t}) p(m_\mathrm{i} \,|\, \gamma_{1:t-1}, x_{1:t})}{p(\gamma_t \,|\, \gamma_{1:t-1}, x_{1:t})} \tag{2.3}$$

With Markov assumption:

$$p(\gamma_t \,|\, x, \gamma_{1:t-1}) = p(\gamma_t \,|\, x)$$

Bayes rule

$$p(m_\mathrm{i} \,|\, \gamma_{1:t}, x_{1:t}) = \frac{p(\gamma_t \,|\, m_\mathrm{i}, \gamma_{1:t-1}, x_{1:t}) p(m_\mathrm{i} \,|\, \gamma_{1:t-1}, x_{1:t})}{p(\gamma_t \,|\, \gamma_{1:t-1}, x_{1:t})}$$

Markov

$$= \frac{p(\gamma_t \,|\, m_\mathrm{i}, x_t) p(m_\mathrm{i} \,|\, \gamma_{1:t-1}, x_{1:t-1})}{p(\gamma_t \,|\, \gamma_{1:t-1}, x_{1:t})}$$

With the further calculation, the occupancy probability for each cell is gained.

Bayes rule

$$= \frac{p(m_\mathrm{i} \,|\, \gamma_t, x_t) p(\gamma_t \,|\, x_t) p(m_\mathrm{i} \,|\, \gamma_{1:t-1}, x_{1:t-1})}{p(m_\mathrm{i} \,|\, x_t) p(\gamma_t \,|\, \gamma_{1:t-1}, x_{1:t})} \tag{2.4}$$

Markov

$$= \frac{p(m_\mathrm{i} \,|\, \gamma_t, x_t) p(\gamma_t \,|\, x_t) p(m_\mathrm{i} \,|\, \gamma_{1:t-1}, x_{1:t-1})}{p(m_\mathrm{i}) p(\gamma_t \,|\, \gamma_{1:t-1}, x_{1:t})} \tag{2.5}$$

Bayesian estimation can be calculated recursively too.

In the vehicle context if:

n-dimensional vector $X_t = (x_{t_1},\ x_{t_2}, ..., x_{t_n})$ is considered as vehicle pose at given time t.

n-dimensional vector $\gamma_t = (\gamma_{t_1},\ \gamma_{t_2},\ ..., \gamma_{t_n})$ be the sensor measurement at given time t from a sensor S.

As it is mentioned the main goal of the occupancy grid algorithm is to estimate:

$$p(m \,|\, \gamma_{1:t}, x_{1:t})$$

It is the probability that the grid is in a specific state when each vehicle position is determined and all measurements are given. if a grid has n binary cells 2^n possible configurations for the grid which is not easy to deal with it computationally.

To simplify the problem and according to Bayes theorem, it is assumed that each cell is independent of the other cells. if several cells are occupied by an object, they will be dependent and they all depends to that object. As it is mentioned before the probability of each cell is calculated by binary Bayes filter rule[11]:

$$p(m_{\mathrm{i}} \,|\, \gamma_{1:t}, x_{1:t}) = \frac{p(m_{\mathrm{i}} \,|\, \gamma_t, x_t)p(\gamma_t \,|\, x_t)p(m_{\mathrm{i}} \,|\, \gamma_{1:t-1}, x_{1:t-1})}{p(m_{\mathrm{i}})p(\gamma_t \,|\, \gamma_{1:t-1}, x_{1:t})}$$

Above rule is defined for opposite event too which is:

$$p(\neg m_{\mathrm{i}} \,|\, \gamma_{1:t}, x_{1:t}) = \frac{p(\neg m_{\mathrm{i}} \,|\, \gamma_t, x_t)p(\gamma_t \,|\, x_t)p(\neg m_{\mathrm{i}} \,|\, \gamma_{1:t-1}, x_{1:t-1})}{p(\neg m_{\mathrm{i}})p(\gamma_t \,|\, \gamma_{1:t-1}, x_{1:t})} \qquad (2.6)$$

According to the probability:

$$p(\neg m_{\mathrm{i}} \,|\, \gamma_{1:t}, x_{1:t}) = 1 - p(m_{\mathrm{i}} \,|\, \gamma_{1:t}, x_{1:t})$$

As it is mentioned, the probability of the entire grid (map) is calculated by:

$$p(m \,|\, \gamma_{1:t}, x_{1:t}) = \Pi p(m_{\mathrm{i}} \,|\, \gamma_{1:t}, x_{1:t})$$

A recursive formula for updating rule of occupancy grid [11]:

With $p(m_{\mathrm{i}} \,|\, \gamma_{1:t}, x_{1:t})$ divide by $p(\neg m_{\mathrm{i}} \,|\, \gamma_{1:t}, x_{1:t})$ and applying the logarithm:

$$V_t(m_{\mathrm{i}}) = \log\left(\frac{p(m_{\mathrm{i}} \,|\, x_{1:t}, \gamma_{1:t})}{1 - p(m_{\mathrm{i}} \,|\, x_{1:t}, \gamma_{1:t})}\right) = V_{t-1}(m_{\mathrm{i}}) + \log\left(\frac{p(m_{\mathrm{i}} \,|\, x_t, \gamma_t)}{1 - p(m_{\mathrm{i}} \,|\, x_t, \gamma_t)}\right) - \log\left(\frac{p(m_{\mathrm{i}})}{1 - p(m_{\mathrm{i}})}\right) \quad (2.7)$$

The formula (2.7) states each grid cell update in each iteration.

Where

$V_t(m_{\mathrm{i}})$: is the current cell grid map value.

$V_{t-1}(m_{\mathrm{i}})$: is the previous cells grid map value.

$\log\left(\frac{p(m_{\mathrm{i}} \,|\, x_t \,,\, \gamma_t)}{1 - p(m_{\mathrm{i}} \,|\, x_t, \gamma_t)}\right)$: is inverse sensor model that indicate the distribution of current measurements.

$\log\left(\frac{p(m_{\mathrm{i}})}{1 - p(m_{\mathrm{i}})}\right)$: is the prior probability. it can be considered as initialization.

The update rule formula can be used for static object. Because all cells are unknown at beginning, the constant $p(m_{\mathrm{i}}) = 0.5$.

Switching cells makes a problem for moving objects due to not building up a confidence. To solve the issue a decay factor can be used after the update rule and the below formula is applied:

$$p(m_{\mathrm{i}}) \;\leftarrow\; (p(m_{\mathrm{i}} - 0.5)\mathrm{e}^{\frac{-\Delta t}{T}} + 0.5) \qquad (2.8)$$

where

Δ t: is update interval.

T: is time constant of decay.

The decay factor is the speed of losing previous measurements and going back to the factor that is a measure of how fast the grid forgets old measurements and goes back to the unknown position before having new measurements [11].

In probabilistic approach, for some reasons such as being on boundary area (being partially occupied), there is some numerical fluctuations for cell occupancy probability value near 0 and 1. With applying logarithm, the occupancy probability can be calculated more accurate.

The occupancy probability value for each cell at any t moment is:

$$p(m_{\mathrm{i}} \mid \ x_{1:t} \ , \ \gamma_{1:t}) = 1 - \frac{1}{\mathrm{e}^{V_t(m_{\mathrm{i}})}} \qquad (2.9)$$

2.7 Stereo vision algorithm and occupancy grid

So far, the occupancy grid map is investigated in terms of probabilistic approach. Now, the way of occupancy grid map generation is considered through the stereo vision algorithm. In stereo vision, pixel information of two pictures of an object is extracted from several views (more than one) of a scene at the same time for estimating disparity map [12]. In stereo vision, two perfectly aligned image sensor(camera) along a horizontal or vertical line is required. The captured image by camera needs to be corrected based on stereo-pair process. A processed image is passed through an algorithm. A stereo vision camera has two or more lenses and each lens has own image sensor. Stereo vision systems are able to simulate human binocular vision is simulated by stereo vision system which is able to capture 3D images. The difference between two positions of the cameras is called the disparity and it is used to find out the depth of objects with respect to the camera position. To convert raw data to occupancy grid map through the stereo vision algorithm the robot operating system (ROS) is needed with using various middle wares to deal with localization problem [5], [13].

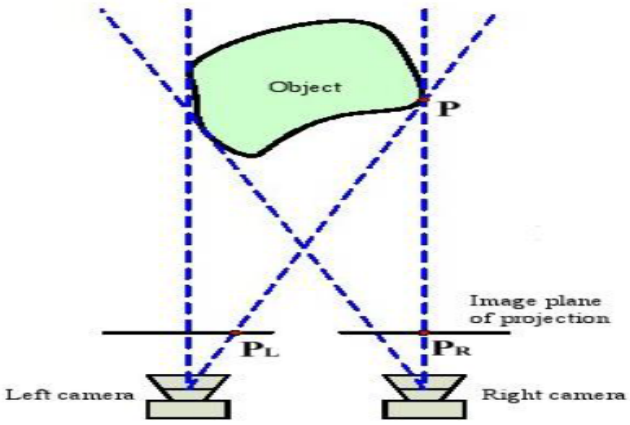

Figure 2.5. Geometrical diagram of stereo vision system [14].

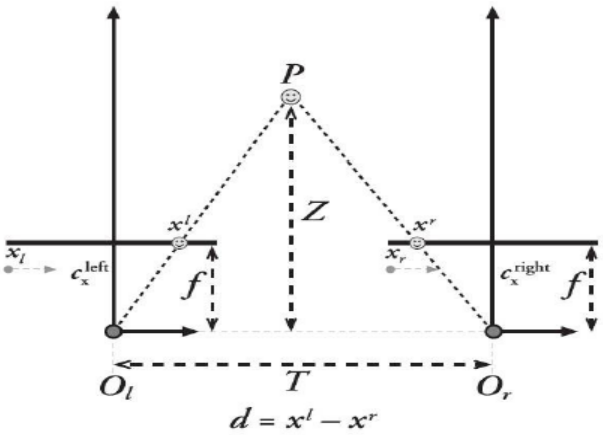

Figure 2.6. Stereo vision system with the disparity of d [13].

P is principle point of both images.

2.7.1 Free space map and obstacle map

As it is mentioned above, one of the essential and most powerful method to detect the free space and obstacle in drivable area and surrounding the ego car is the stereo vision algorithm. In stereo vision, obstacle information and free region in front of the ego car and determining the orientation and position of the camera relation to objects that is known as camera pose estimation is considered as outputs which is used as input for generating

occupancy grid map. Height (h), pitch angle (θ), and roll angle (ρ) are the stereo vision system external parameters which is calculated constantly to find camera position estimation. All calculations are done with respect to the ground. To do prior calibration (finding focal length and some other parameters) the baseline (b), focal length (α) and optic center (u_0,v_0) is needed (Musleh et al., 2014) [5]. The disparity map is generated for free space and obstacle detection calculation by using the proposed method in (Musleh et al., 2012). In this map some information about depth (Z) and disparity (Δ) level is given (cell or pixel values in depth and disparity level are corresponding). uv-disparity is calculated by the method in (Hu et al., 2005). To determine Free space and obstacle maps, another disparity called u-disparity is needed for demonstration of higher value cells which is corresponding to the obstacles. There are two types of cells in terms of color in disparity map which is not constant. All cells in disparity level are corresponding to a disparity value, the higher value is corresponding to white cell belong to u-disparity and representing the obstacle map and rest of the cells representing the free map. The boundary area representing white cell and belong to u-disparity [5].

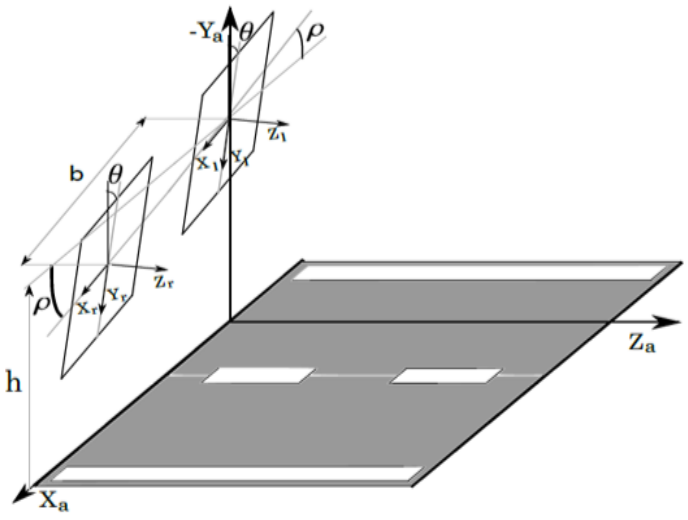

Figure 2.7. Camera position from the ground [15]

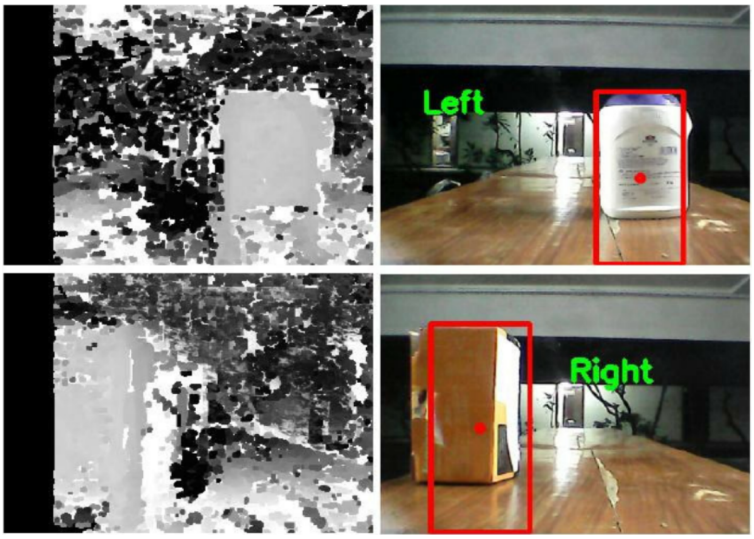

Figure 2.8. Disparity map in an indoor environment of several obstacles in the left side and detected obstacle based on distance in the right side [13].

2.7.2 Occupancy Grid Map generation

In section (2.7.1) is stated obstacle map with the different segmentation is generated and with subtraction of all edges with using the given method in (Musleh et al., 2011), several desired areas (regions of interest) are built. With obtaining the bounding boxes for each area, the obstacle will be managed better and simpler. As is mentioned the obstacle map is determined with a higher disparity value, a disparity value (Δ) is dedicated to each area. With using the below equation with the derivation form in (Musleh et al., 2014), the 3D coordinate for each obstacle cell (pixel) is earned [5].

$$X = (b\cos\theta\sin\rho(v - v_0) + b\cos\rho(u - u_0) + \alpha b\sin\rho\sin\theta)/\Delta$$

$$Y = -h + (b\cos\rho\cos\theta(v - v_0) - b\sin\rho(u - u_0) + \alpha b\cos\rho\sin\theta)/\Delta$$

$$Z = (\alpha b\cos\theta - b\sin\theta(v - v_0))/\Delta \qquad (2.10)$$

There are elevated and on-road obstacles based on boundary boxes. According to the coordinate graph, the elevation is calculated from its bottom left corner in disparity map.

Elevated obstacle turns to on-road obstacle if the elevation value approaches to zero. All cells (pixels) in an obstacle, because is located on the same disparity level, is assumed have same distance to the camera. Each cell in generated grid map based on the grid resolution represent a specific area in the world. Gray level describes the probability of the cell that is being occupied by an obstacle [5].

Table 2.1. Occupancy probability value table [5]

Probability value	Description
0-1 (0-100 by percentage)	Occupancy probability (drivable area)
-1	Unknown (Non-drivable area)

There is an inverse relationship between increasing distance and cell transparency in the stereo-vision system, it means if the distance between camera and target increases, then the cell resolution decreases. Based on the distance, three levels of resolution in occupancy are defined that is called confidence levels.

Level 1: If distance is less than 22 meters, which sensor measurement is reliable and cells resolution are the highest. This area is free space (not occupied), and occupancy probability is zero.

Level 2: If distance is between 22-45 meters, sensor measurement reliability goes down and pixel resolution reduces, and probability of occupancy is 0.14.

Level 3: If distance is greater than 45 meters resolution is lowest and probability of occupancy is 1, means is occupied.

Those points that camera can see them are called grid. To find the x and y coordinates for disparity map, the depth in this point must be maximum and Δ=1.

The grid represents only points that camera's visibility can cover those points. So, to calculate the x and y coordinates of the top corner of disparity map, the maximum depth must be assigned. In maximum depth the Δ=1. Rightmost point and leftmost point of the disparity map is located on the x-axis. From the bottom center of occupancy grid, two lines are drawn to those cells. The part that camera cannot see it is called unknown. Now, obstacle location is found [5].

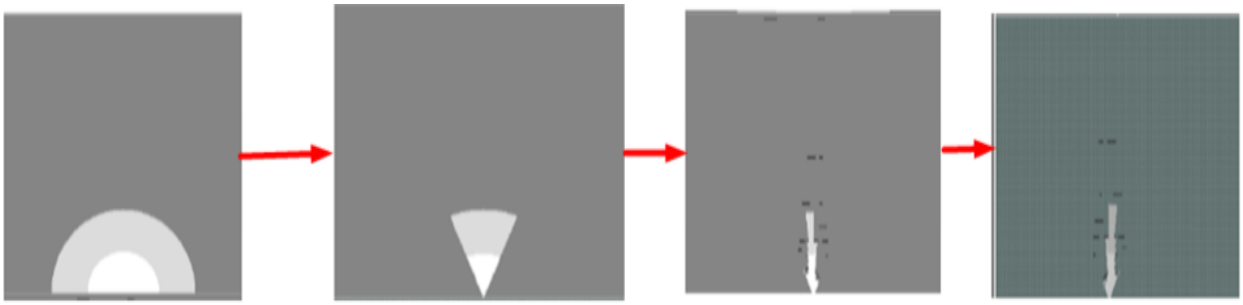

Figure 2.9. Creating process of occupancy map [5]

2.8 Depth calculation by stereo vision geometry

The images produced by stereo vision algorithm are 2D images that are representing corresponding real 3D images which is captured from more than one camera. Despite being 2D, stereo image pairs, carry more information of real world compare to standard 2D images such as scene and depth [16]. When a 3D structure is extracted stereo image pairs, some calculation such as image rectification is accomplished. This process is called computational stereo. In stereo image pairs matching the same points is called corresponding problem. In camera Geometry, with image rectification, 2D search could be simplified to one dimensional search for corresponding points. This is based on epipolar rectification [16].

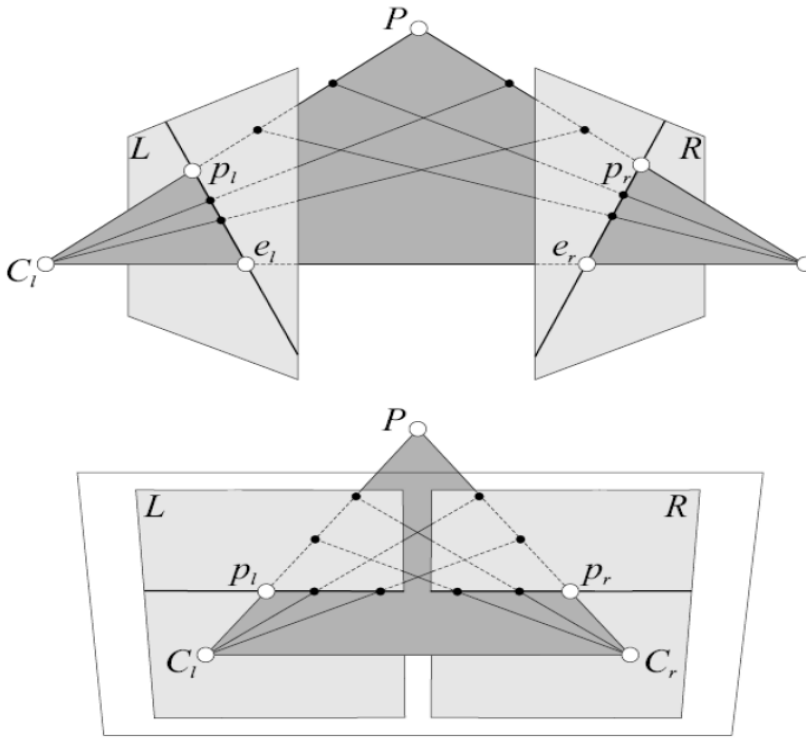

Figure 2.10. Epipolar geometry or rectification [16].

Depth (z) Calculation:

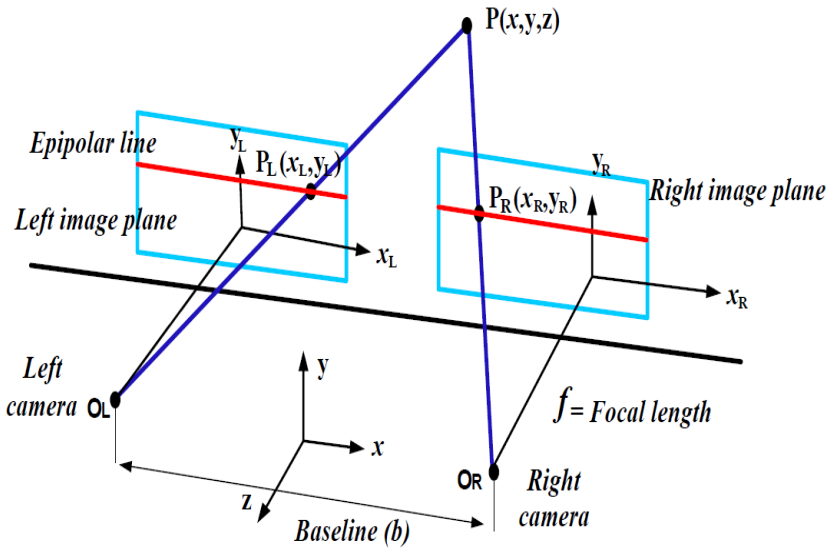

Figure 2.11. Epipolar geometry or rectification [14].

With triangular similarity:

$$\frac{x_L}{f} = \frac{x + \frac{b}{2}}{z}$$

$$\frac{x_R}{f} = \frac{x - \frac{b}{2}}{z}$$

$$z = \frac{bf}{x_L - x_R} = \frac{bf}{d} \qquad (2.10)$$

Where

P: is object position

O_L: is left camera center

O_R: is right camera center

b: is the base line

P_L: is projection of P in left image plane

P_R: is projection of P in right image plane

f: is the focal length

$d = x_L - x_R$: is disparity

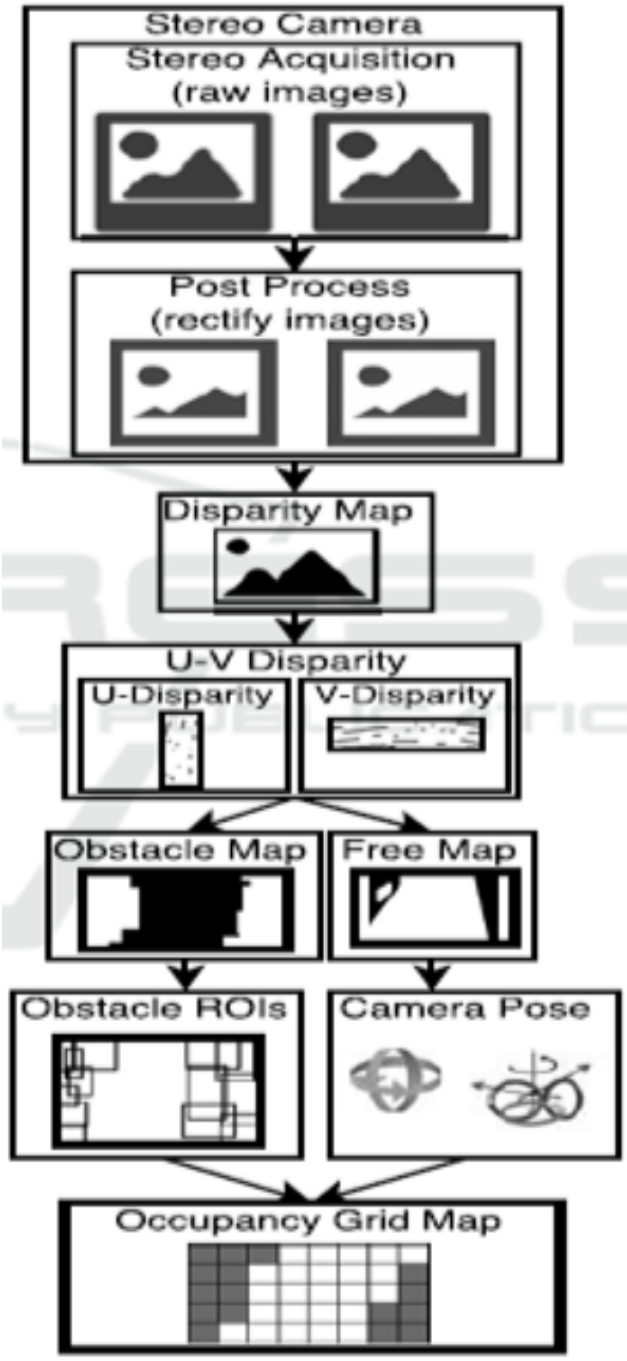

Figure 2.12. Generating OGM by ROS [5].

Generating 2D occupancy grid map

```
prompt = 'What is the original value? ';
wmap = input('initial wmap value:');   %creating empty map with width, height, and resolution
hmap = input('initial hmap value:');
rmap = input('initial rmap value:');
xav = input('initial xav value:');     %vehicle pose xav,yav,zav
yav = input('initial yav value:');
zav = input('initial zav value:');  %in 2D z=0
rx = input('initial rx value:');   %range entry
ry = input('initial ry value:');
%angle entry
%lower angle
al = input('initial al value:');
%upper angle
au = input('initial au value:');
mr = input('initial mr value:');   %maxrange
k = input('initial k value:');     %coefficient
x = input('initial x value:');    %occupancy in specific point
y = input('initial y value:');
map = occupancyMap(wmap,hmap,rmap); %creating empty map
pose = [xav,yav,zav]; %vehicle pose
ranges = k*ones(rx,ry);  %range
angles = linspace(al,au,rx);  %angle
maxrange = mr;  %max range
%Create a lidarScan object with the specified ranges and angles.
scan = lidarScan(ranges,angles);
%Insert the laser scan data into the occupancy map.
insertRay(map,pose,scan,maxrange);
%Show the map to see the results of inserting the laser scan.
show(map)
%Check the occupancy of the spot directly in front of the vehicle.
getOccupancy(map,[x y])
%Add a second reading and view the update to the occupancy values.
%The additional reading increases the confidence in the readings.
%The free and occupied values become more distinct.
insertRay(map,pose,scan,maxrange);
show(map)
```

Figure 2.13. Generating 2D OGM algorithm.

A 2D occupancy grid map object is generated. There is a probability value for each cell in the map. The values close to 1 (high probability) represent the corresponding cell is occupied by obstacle and values close to 0 represent the related cell is free. Occupancy map is used in navigation algorithm such as path planning, mapping application for collision-free path and etc. [17].

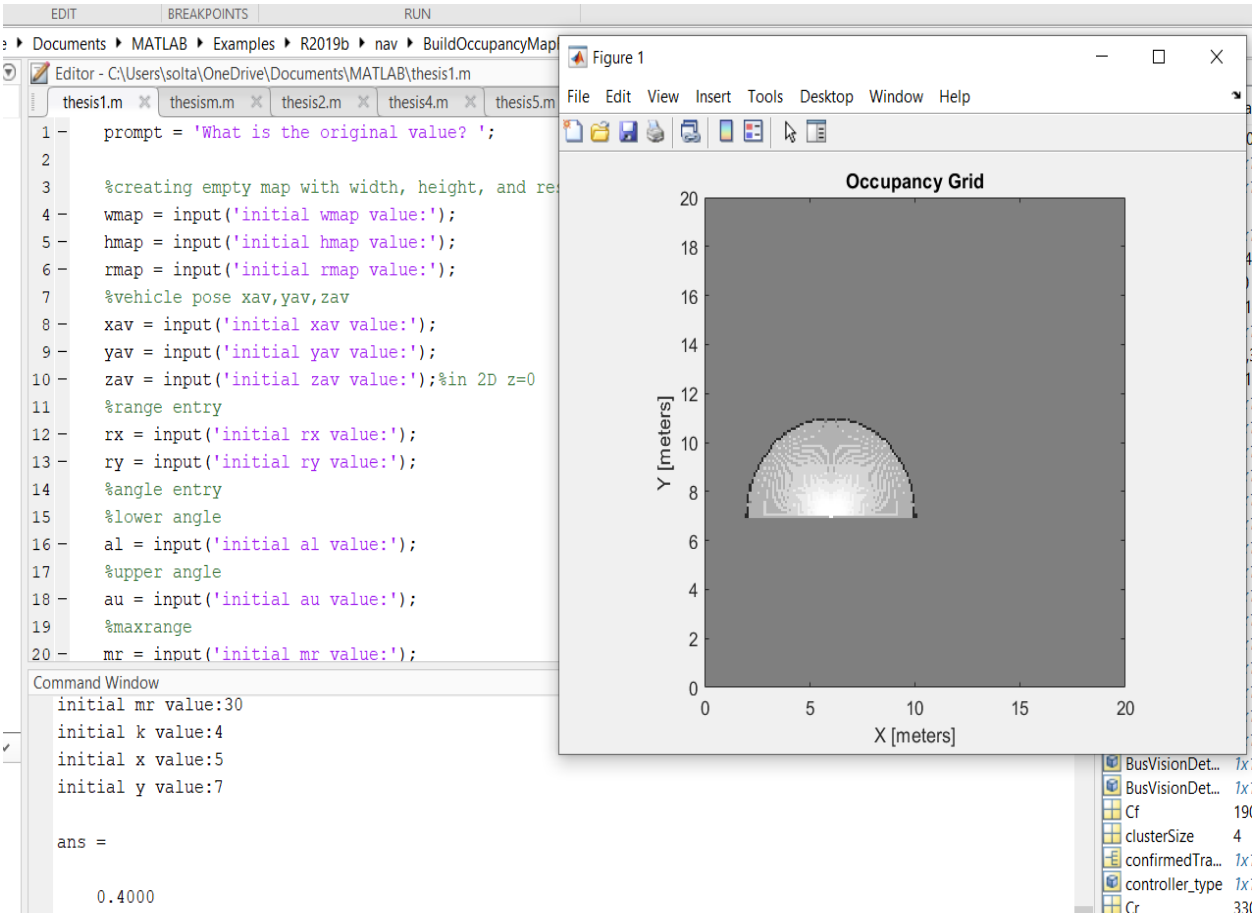

Figure 2.14. Generating occupancy grid with inserting laser scan in MATLAB .

Above figure states the probability occupancy at point (5,7) in front of the car is 0.4.

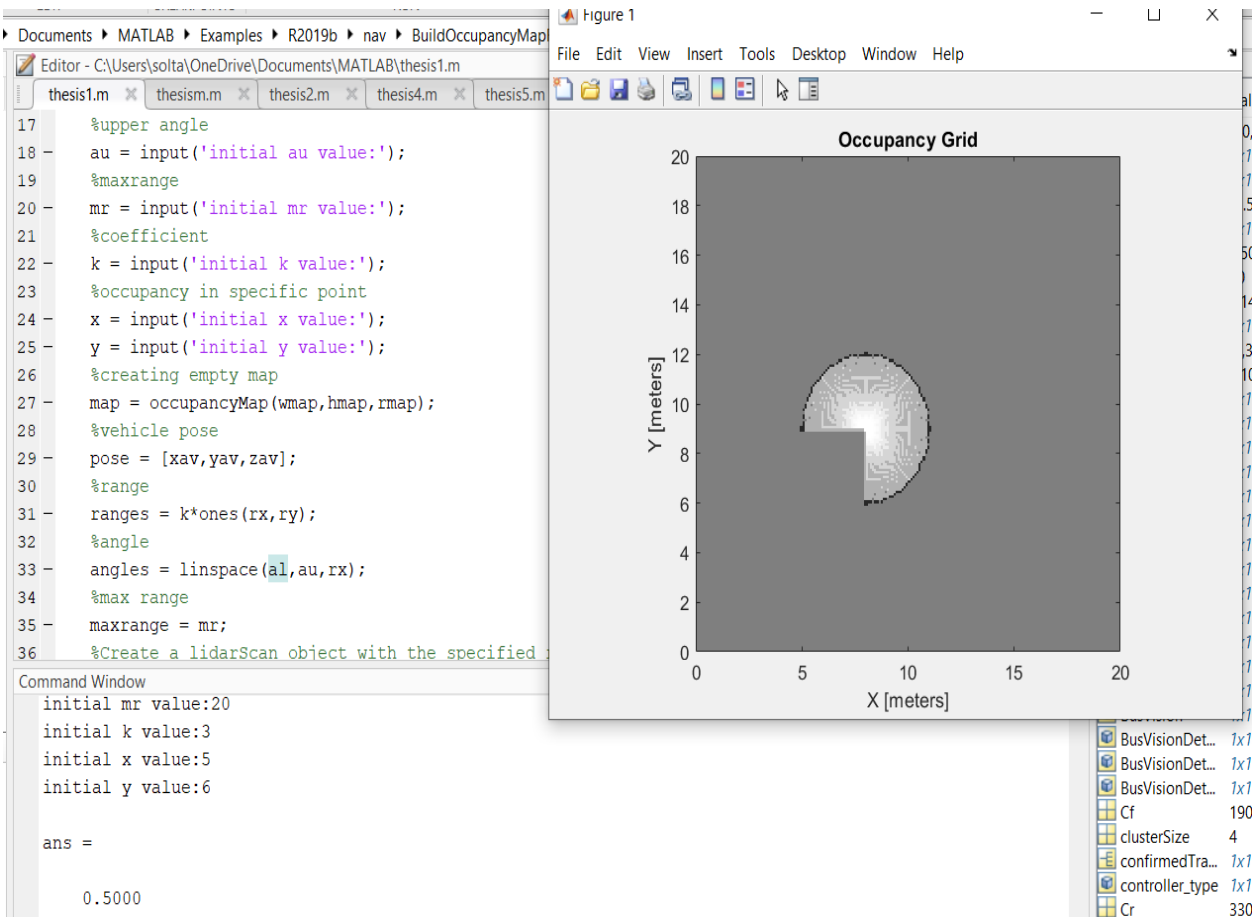

Figure 2.15. Generating occupancy grid with inserting laser scan in MATLAB .

Above figure states the probability occupancy at point (5,6) in front of the car is 0.5.

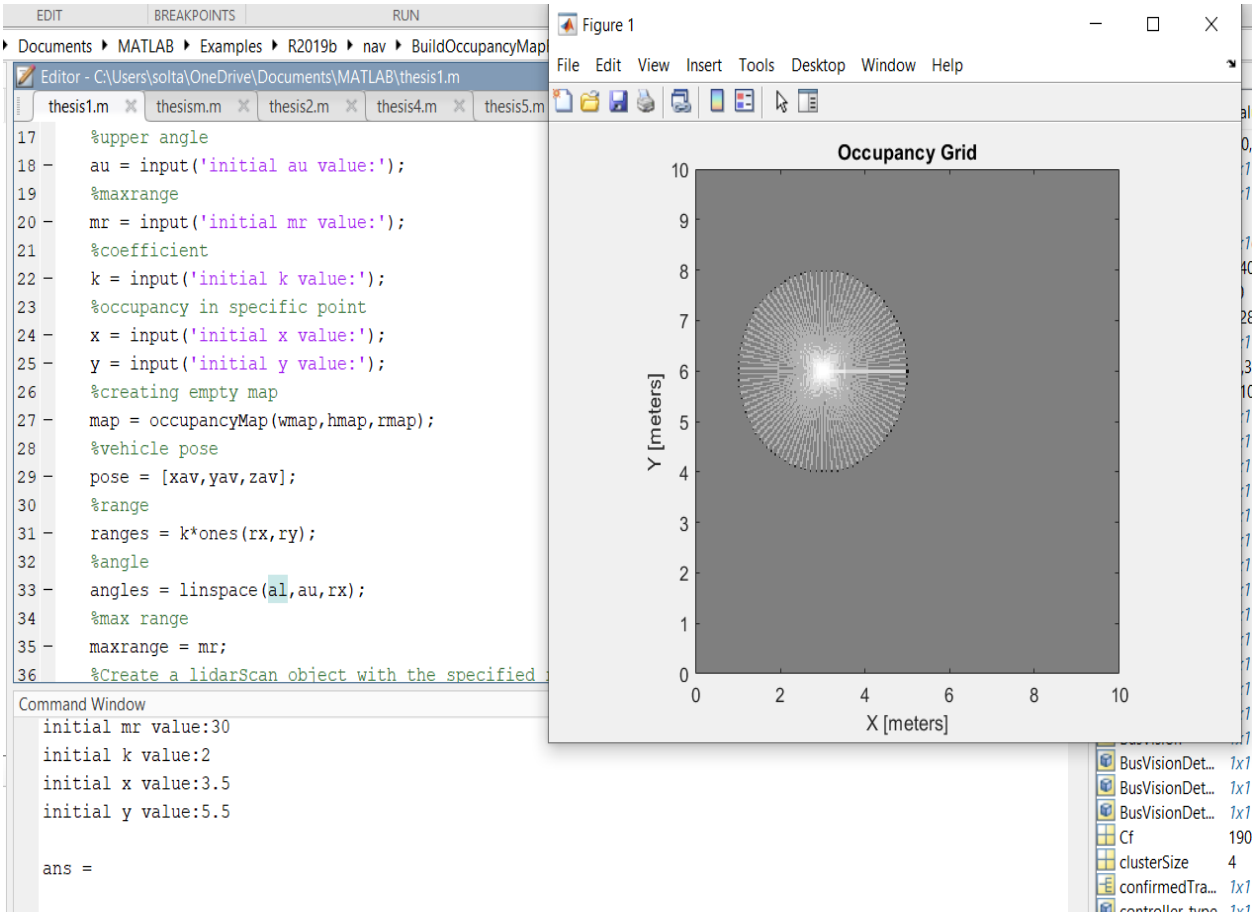

Figure 2.16. Generating occupancy grid with inserting laser scan in MATLAB .

Above figure states the probability occupancy at point (3.5,5.5) in front of the car is 0.4.

An empty occupancy grid map is generated, then, vehicle pose, angles, ranges and max range for laser scan are given. Laidar scan object with the angles and ranges are given. With the inserting the laser scan data into occupancy map, the occupancy probability of cell around the AV is earned.

The following is a 3D OGM generation algorithm.

```
r = input('initial r value:'); %cell's resulution
xs = input('initial xs value:');  %sensor poses
ys = input('initial ys value:');
zs = input('initial zs value:');
qws = input('initial qws value:');
qxs = input('initial qxs value:');
qys = input('initial qys value:');
qzs = input('initial qzs value:');
%first point
xpl1 = input('initial1 xpl value:'); %leftside value of x
d1 = input('initial d1 value:'); %distance
xpr1 = input('initial xpr1 value:'); %rightside value of x
yp1 = input('initial yp1 value:'); %y value of point
zp1 = input('initial zp1 value:'); %z value of point
%second point
xpl2 = input('initial1 xpl2 value:'); %leftside value of x
dx = input('initial dx value:'); %distance
xpr2 = input('initial xpr2 value:'); %rightside value of x
ypl2 = input('initial1 ypl2 value:'); %leftside value of y
dy = input('initial dy value:'); %distance
ypr2 = input('initial ypr2 value:'); %rightside value of y
zpl2 = input('initial1 zpl2 value:'); %leftside value of z
dz = input('initial dz value:'); %distance
zpr2 = input('initial zpr2 value:'); %rightside value of z
mr = input('initial mr value:'); %max range
vr = input('initial vr value:'); %vehicle radius
sr = input('initial sr value:'); %safety radius
%Create an occupancyMap3D object with a map
%resolution of r cells/meter
map3D = occupancyMap3D(r);
% Define a set of 3-D points as an observation
% from a pose [xs ys zs qws qxs qys qzs]. This pose is for the
% sensor that observes these points and is centered on
% the origin. Define two sets of points to insert multiple
%observations.
pose = [ xs ys zs qws qxs qys qzs];
points1 = repmat((xpl1:d1:xpr1)', yp1, zp1);
points2 = [(xpl2:dx:xpr2)' (ypl2:dy:ypr2)' (zpl2:dz:zpr2)'];
maxRange = mr;
insertPointCloud(map3D,pose,points1,maxRange)
show(map3D)
insertPointCloud(map3D,pose,points2,maxRange)
show(map3D)
vehicleRadius = vr;
safetyRadius = sr;
inflationRadius = vehicleRadius + safetyRadius;
inflate(map3D, inflationRadius);
show(map3D)
```

Figure 2.17. Generating 3D occupancy grid map algorithm in MATLAB .

The sensors are used to map the surrounding and an obstacle will be stored in the *occupancyMap3D* object. With adding the points in the created map, the obstacle will be identified. The map will be inflated for adding a buffer zone around the obstacle for safe operation [18].

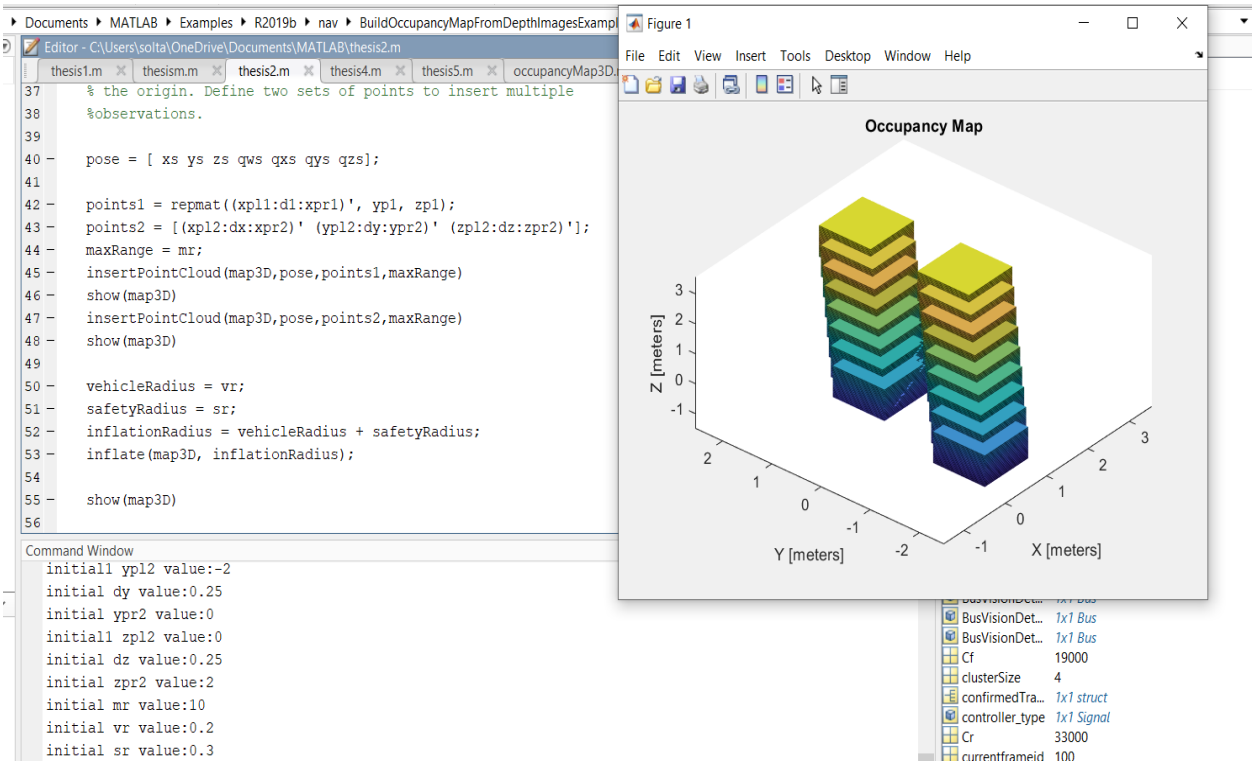

Figure 2.18. Generating 3D occupancy grid map in MATLAB .

2.9 Inflation of obstacles

In occupancy grids, the obstacles(coordinates) can be inflated. In inflating obstacles, a safety factor is added on obstacles for creating a buffer zone between the AV or robot and obstacle in surrounding environment. A specified radius is converted to the number of cells by the inflate function. Each occupancy grid object has inflate function. By multiplying the resolution and radius value, the number of cells will be earned that must be rounded up [19].

First, a 2D empty map with desired size is designed and then a desired occupancy grid with an arbitrary position is generated which is obstacle location. Each occupied cell is inflated by adding occupied space around each point. This algorithm shows how to create an arbitrary map, determine the obstacle locations and inflate it by a radius of n meter. Extra plots on the figure help illustrate the inflation and shifting due to conversion to grid locations.

(xoc, yoc) is the grid position in the empty map. r is the number of cells(black cell) (r=1 create one black square , r=2 create 2 blacks square etc.). $rinf$ changes the size grid(black part). if the r and $rinf$ have bigger value it will be more accurate.

```
prompt = 'What is the original value? ';
%generattion 2D plane with desired sizes
xmap = input('initial xmap value:');
ymap = input('initial ymap value:');
%Resolution for 2D map(cell resolution per meter)
r = input('initial r value:');
%position of binary occupancy and thershold must be defined
% and they can not be greater than xmap and ymap
xoc = input('initial xoc value:');
yoc = input('initial yoc value:');
%radius of occupancy
roc = input('initial roc value:');
%inflation value(inflation radius
rinf = input('initial rinf value:');
%show(map)
map = binaryOccupancyMap(xmap,ymap,r);
setOccupancy(map,[xoc yoc], roc);
inflate(map,rinf);
show(map)
A=[xmap  ymap];
M=[r  rinf];
hold on
theta = linspace(0,2*pi);
x = xoc-0.1+min(M)*cos(theta); % x circle coordinates
y = yoc-0.1+min(M)*sin(theta); % y circle coordinates
plot(xoc,yoc,'*g','MarkerSize',mean(A)) % Original location
plot(xoc-0.1,yoc-0.1,'xr','MarkerSize',mean(A)) % Grid location center
plot(x,y,'-b','LineWidth',2*min(M)); % Circle of radius rinf m.
axis([0 xmap 0 ymap])
ax = gca;
ax.XTick = (0-xoc:1:xmap+xoc);
ax.YTick = (0-yoc:1:ymap+yoc);
grid on
legend('Original Location','Grid Center','Inflation')
```

Figure 2.19. Inflation of obstacle MATLAB code .

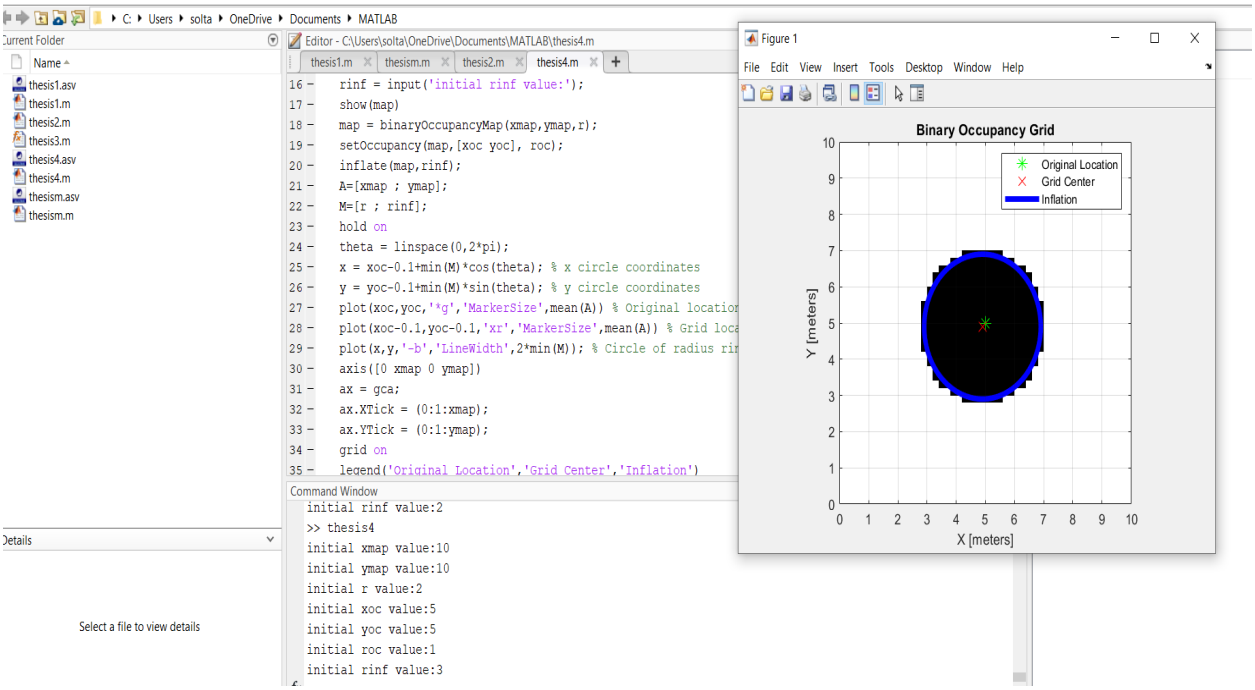

Figure 2.20. Inflation of obstacle at point (5, 5).

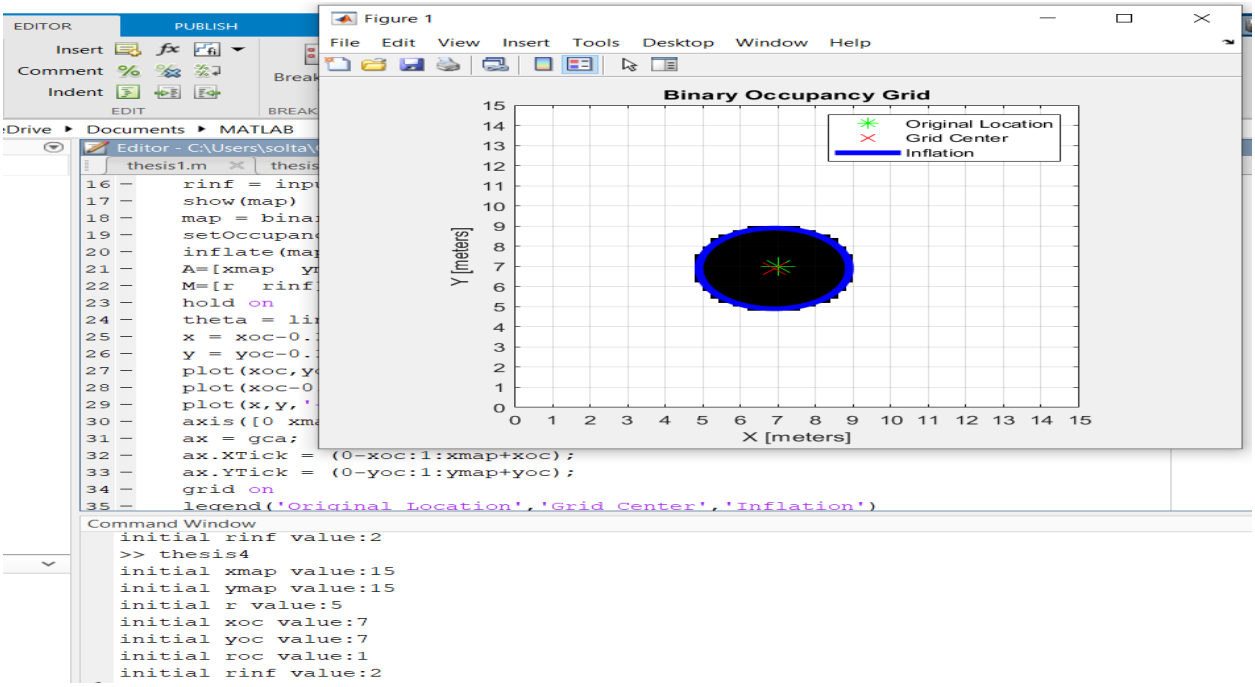

Figure 2.21. Inflation of obstacle at point (7, 7).

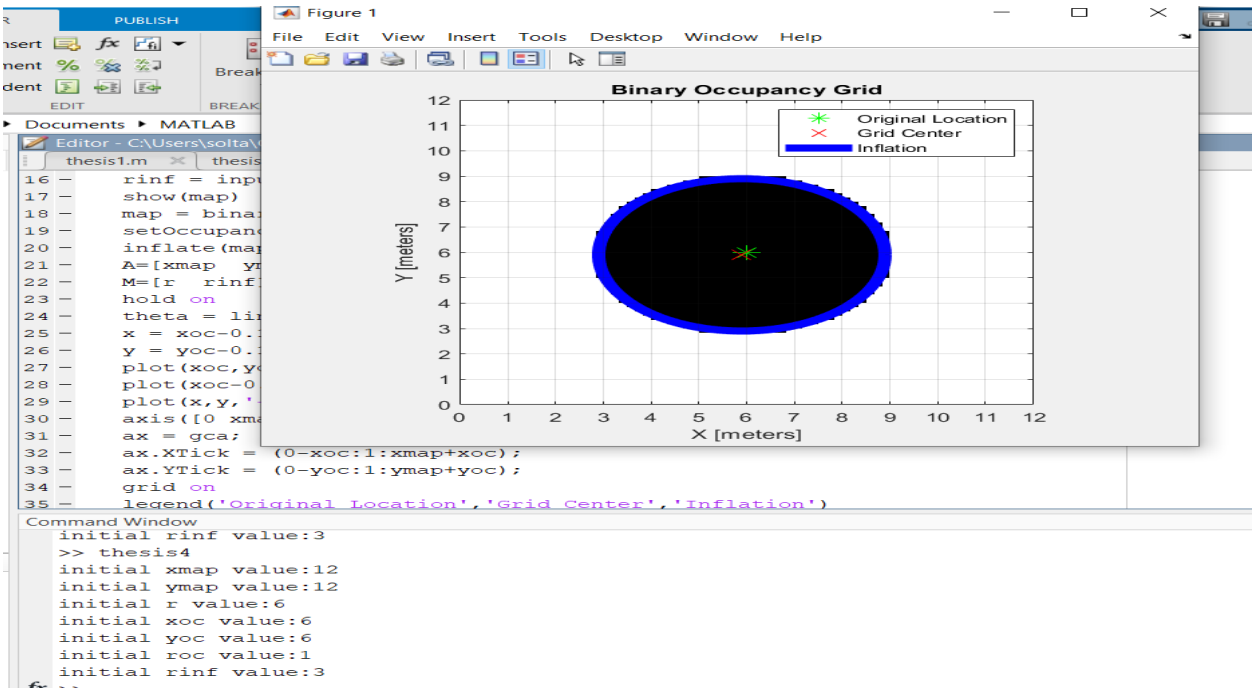

Figure 2.22. Inflation of obstacle at point (6, 6).

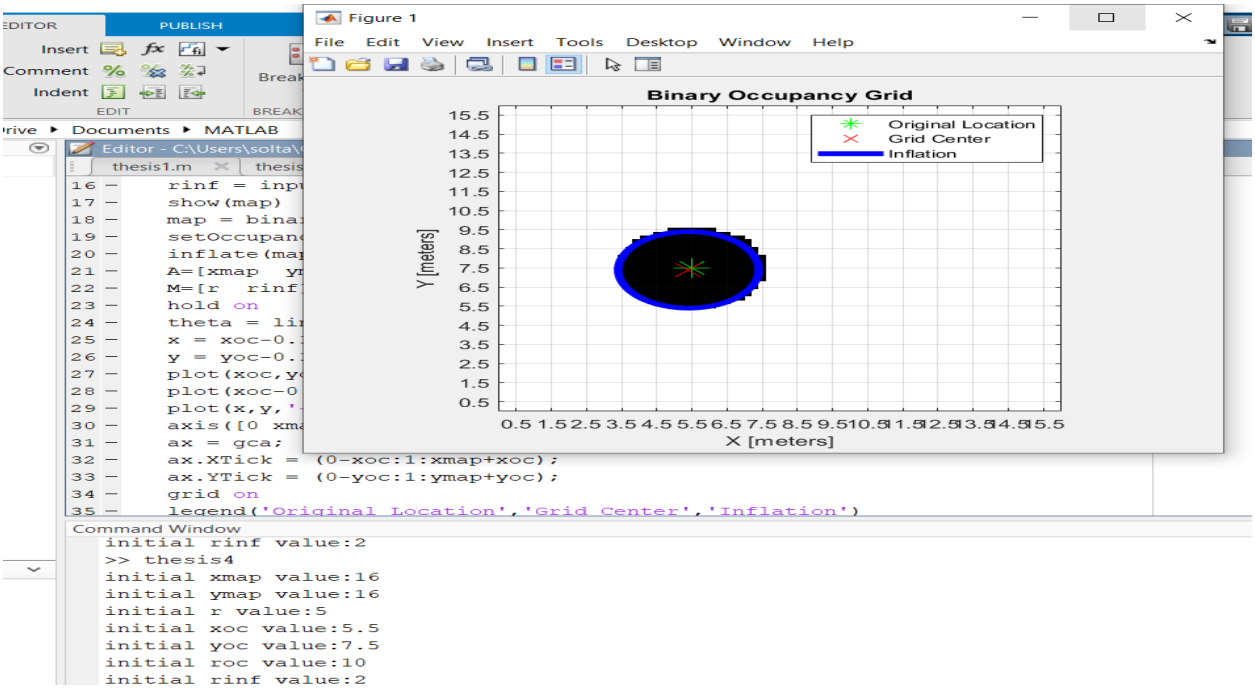

Figure 2.23. Inflation of obstacle at point (5.5, 7.5).

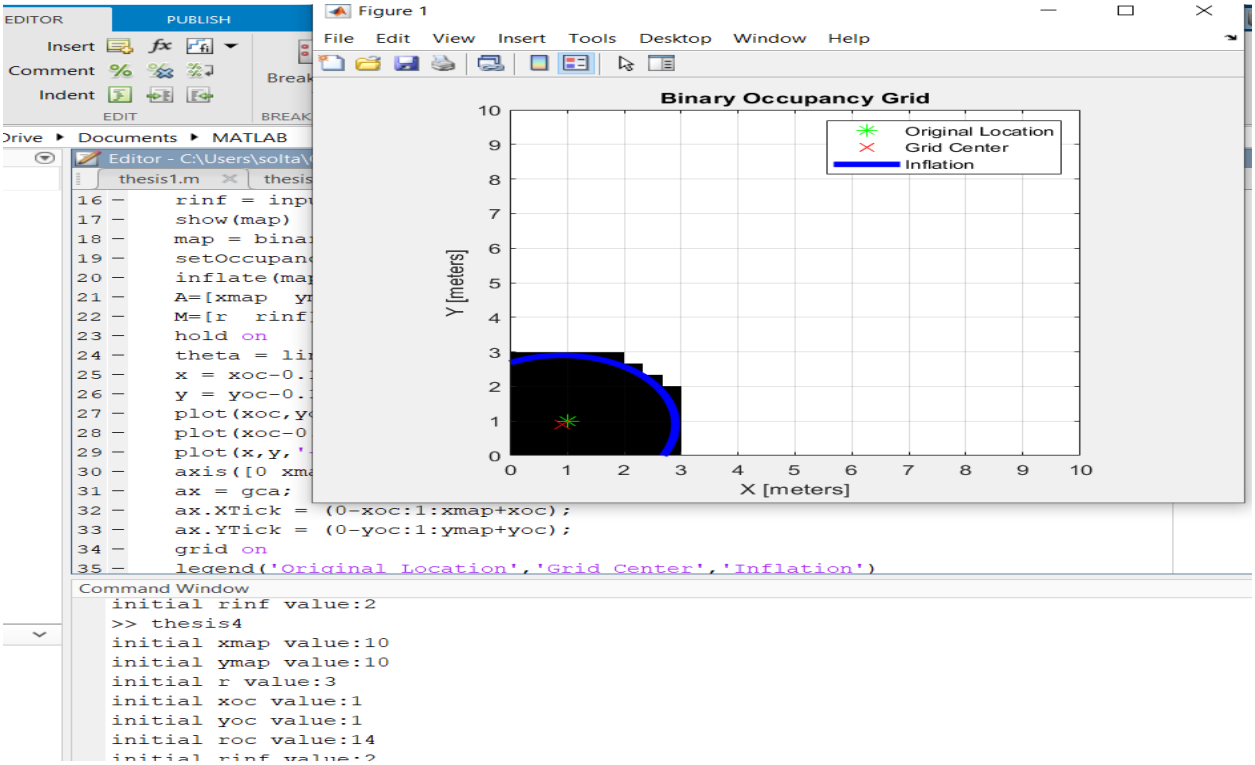

Figure 2.24. Inflation of obstacle at point (1, 1).

Stereo vision is strong tool for depth estimation, obstacle detection and generating occupancy grid map. However the same as any software-defied tools is restricted. Different algorithms have been used that they have different computational cost and time complexity that generate different quality disparity map with different transparency and accuracy. The surrounding is occupied by various waves, which interfere with the map disparity generation. In this case, the algorithm with optimal cost function must be designed to minimize the interference.

3. COLLISION AVOIDANCE

Having a full autonomous car without a collision avoidance system (CAS) is wasting a lot of time and investment. Collision avoidance design must be a major focus for researchers to produce a reliable car with more safety. So, CAS as an advanced safety technology plays a significant role in autonomous vehicle technology. Designing a sophisticated (complex) CAS requires an equipped laboratory with experienced technical staff because of the importance of issue. They must be tested in different conditions to pass the maximum safety test and be send to the market [20]. The ultimate goal is preventing collision or mitigating the severity of collision before it happens.

3.1 Types of collision avoidance system

There are four common collision avoidance system (CAS) between many of them that a fleet can apply to raise the safety which are [20]:

- Forward collision warning system: It control both the AV's speed and the vehicle ahead of it and keep the safety distance. Send alert in case of rear vehicle get too close to the car.
- Lane departure warning system: It warns if the AV is drifting out of its lane.
- Pedestrian detection system: It detects moving objects such as pedestrian and cyclist with sensors help and alert the driver for safety reason.
- Automatic braking system: In case of detecting an object near to car by a sensor AV's braking system activate automatically.

CAS warns the driver to the risk by a light, a sound, or both. Visual alert of obstacle is provided by the cameras in different locations in the car [21].

Figure 3.1. Types of collision avoidance system [21].

3.2 Activation

The first action is needed to be taken from the AV to avoid the collision or mitigate based on risk and the number of collision (sequence) is activating of CAS. With using this system, risk of collision is determined, and the future condition is predicted based on AV's surrounding condition. There is some procedure and condition for CAS activation for observing and controlling ego car's surrounding always. For this reason, a predesignated risk boundary (threshold) is given. As soon as car passed the risk boundary, its active algorithm is voided and is replaced by collision avoidance algorithm and the car path is planned for future. The activation time for CAS algorithm is given by following Equation [22]:

$$t_{CAS}\text{=n. t} \quad \text{where} \quad t_0 \leq t_{CAS} \leq t_f \quad \rightarrow \quad 0 \leq t_{CAS} \leq t_f \qquad (3.1)$$

Where

t_{CAS}: is the CAS activation time.

t_0: is the CAS initial (start of algorithm) time and it can be considered zero.

t_f: is the CAS final time (end of CAS algorithm activation).

n: is the iteration number in the CAS algorithm.

t: is the period of each step.

It is assumed all required information is given by sensors.

CAS algorithm stops when $t_{CAS} = t_f$ and after that CAS will be deactivated and AV back to the general algorithm. CAS remains active as long as the path risk is greater than risk threshold [23].

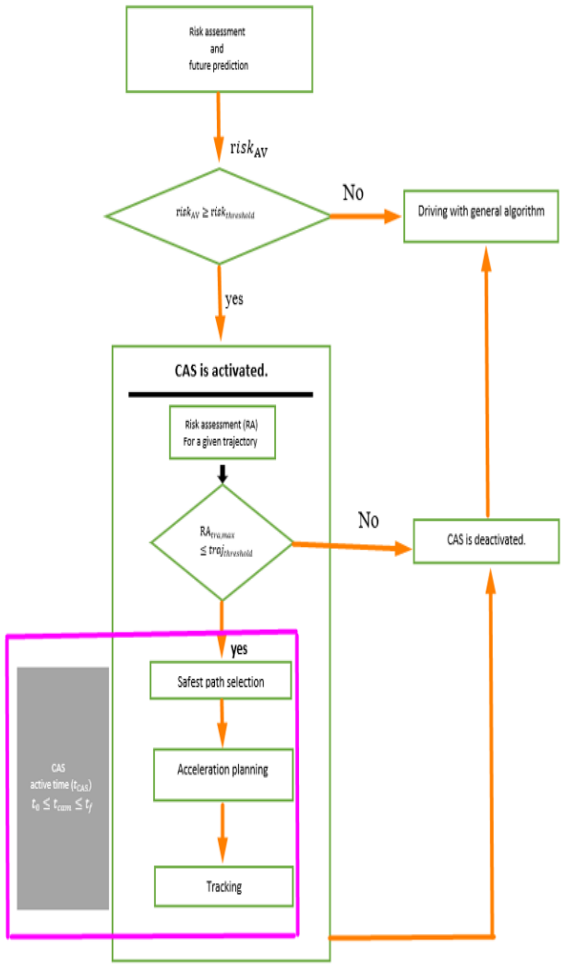

Figure 3.2. Flow chart of AV general driving algorithm and CAS driving algorithm mode change for a candidate trajectory [22].

3.3 Generating map for predictive occupancy

Having a reliable CAS is not sufficient on preventing the collision. It is needed to all information about surrounding risks in terms of time and place be collected. As it mentioned, CAS generate the AV risk and surrounding risk map and as soon as car reaches to risk threshold, a safe velocity and a durable (physically) path must be planned simultaneously. For a moving AV in a candidate path, moving cars, stationary cars, road signs, and other obstacles is a source of risk, having a reliable coordinate system helps to have perfect and precise setting. Supposedly, an AV is considered base coordinate system for surrounding objects like a moving vehicle, predictive occupancy map with using the moving object position, velocity, and acceleration, helps us to predict the AV future in terms of time. With the sensor assistance, AV's driving information is obtained for most accurate risk assessment as follows [22]:

$$D_{AV} = [\ p_{AV,x}, p_{AV,y}, v_{AV,x}, v_{AV,y}, a_{AV,x}, a_{AV,y}\ ] \qquad (3.2)$$

Where

D_{AV}: is the set of AV's driving information.

p_{AV}: is the AV position.

v_{AV}: is the AV velocity.

a_{AV}: is the AV acceleration.

x and y are the coordinate axis, respectively [22].

3.4 Prediction of obstacle position

In CAS technology, prediction of the obstacle position is a big challenge for researchers because of their multiplicity in drivable paths. Here, a method is presented to predict obstacle position at the time of the AV collision with an obstacle. Supposedly, the required information is given by sensing system such as Lidar. In this method, it is assumed the obstacle moves along the y-axis direction and crosses the road, where the AV motion direction

is along the x-axis, so the prediction is valid for y-direction, assuming the obstacle moves at a constant velocity [24]. Its future velocity at certain time t_1 right after time t_0 is calculated by following Equation:

$$v_{ob} = \frac{y_{ob}(t_1) - y_{ob}(t_0)}{\Delta t} \quad where \quad \Delta t = t_1 - t_0 \qquad (3.3)$$

Where

v_{ob} : is obstacle velocity in time $t_0 \leq t \leq t_1$

$y_{ob}(t_1)$: is y coordinate of obstacle at time t_1

$y_{ob}(t_0)$: is y coordinate of obstacle at time t_0

It can be assumed $t_0 = 0$

The obstacle position at time t_1 when the AV collide with the obstacle is calculated in two different position:

$$p_1 = \frac{x_{ob} - x_{AV} - d_{cg}}{v} \qquad (3.4)$$

if the AV is driving on constant velocity.

and

$$p_2 = \frac{-v + \sqrt{v^2 + 2a_x(x_{ob} - x_{AV} - d_{cg})}}{a_x} \qquad (3.5)$$

if the AV brakes and Decelerate with the max deceleration a_x.

where

x_{AV}: is x coordinate of AV in timet_1.

x_{ob} is obstacle x coordinate in time t_1.

d_{cg}: is distance from center of gravity of AV and front end of AV.

a_x: is the AV deceleration.

v: AV velocity at time t_1.

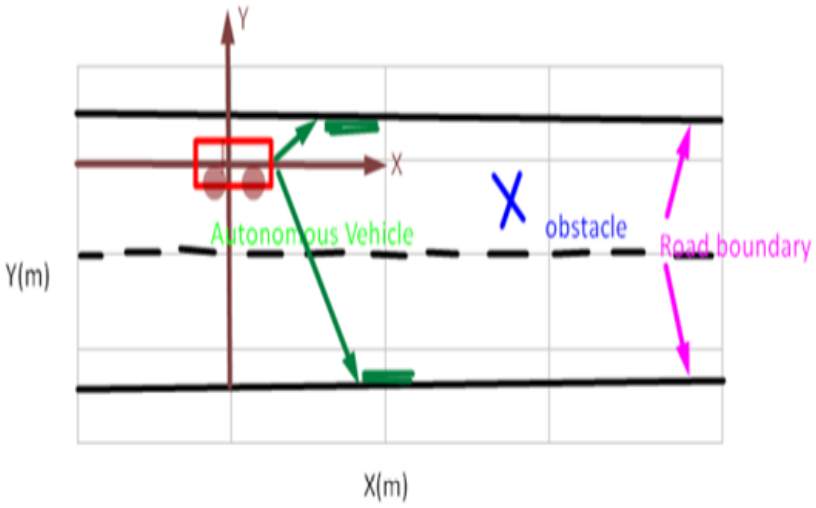

Figure 3.3. Obstacle prediction along y-axis.

3.5 Object tracking

To accurately predict an obstacle, it must be properly tracked. While an obstacle is tracked, three main parameters of it (position, velocity and acceleration) is estimated. Three main factors must be considered when an object is tracked, a frame or coordinate, tracking movement in different frame and collecting data and grouping them. With grouped data in each frame, the velocity and position can be calculated. With knowing position change in an object (obstacle) between each frame, a model can be defined to estimate position, velocity, acceleration. It is assumed the obstacle is dynamic and velocity is constant, which is not in reality. To solve the velocity change issue, Random noise is added. The model for this movement is:

$$\dot{x} = v_x.cos(\theta) + R_x \qquad (3.6a)$$

$$\dot{y} = v_y.sin(\theta) + R_y \qquad (3.6b)$$

$$\Delta\theta = \frac{\Delta s}{r}, \qquad \omega = \frac{\Delta\theta}{\Delta t}, \qquad v = \frac{\Delta s}{\Delta t}, \qquad v = r\omega$$

Because of constant velocity, the acceleration will be:

$$\dot{v_x} = 0 + \dot{R}_x = R_{v_x}$$

$$\dot{v_y} = 0 + \dot{R}_y = R_{v_y}$$

$$\dot{\theta} = R_\theta$$

Where

R_x: is random noise along x-axis for velocity

R_y: is random noise along y-axis for velocity

R_v: is random noise for acceleration

R_θ : is angular acceleration

Random noise or Gaussian noise is a probability distribution that has probability density function same as normal distribution.

Object tracking by Kalman filter

In this section, the Kalman filter is introduced first, then its algorithm is implemented on tracking of an object. The Kalman filter is an optimal estimation algorithm. It is used to estimate a system state that cannot be measured directly, by combining the measurement from different sources. Therefore, the Kalman filter is used to optimal the estimate the variables of interest when they cannot be measured directly but the indirect measurement is available. It is a good tool to find the best estimates of states by combining the measurements from various sensors in the presence of noise. The Kalman filtering is the method to design the optimal state observers. State observation helps to estimates something that is not visible or cannot be measured directly like lots of estimations are being done in Mars. The estimated state x is shown with $(\hat{x})$ [25].

If the real state-space system is given.

$$\dot{x} = Ax + Bu$$

$$y = Cx \qquad (3.7)$$

Where

u is input, y is output, and x is the state that needs to be estimated.

The state observer is a mathematical model that is defined as:

$$\dot{\hat{x}} = A\hat{x} + Bu$$

$$\hat{y} = C\hat{x} \qquad (3.8)$$

If the input of the real system pass through the mathematical model, the estimate of output is earned. The internal state of the system is calculated too ($\hat{x}$).

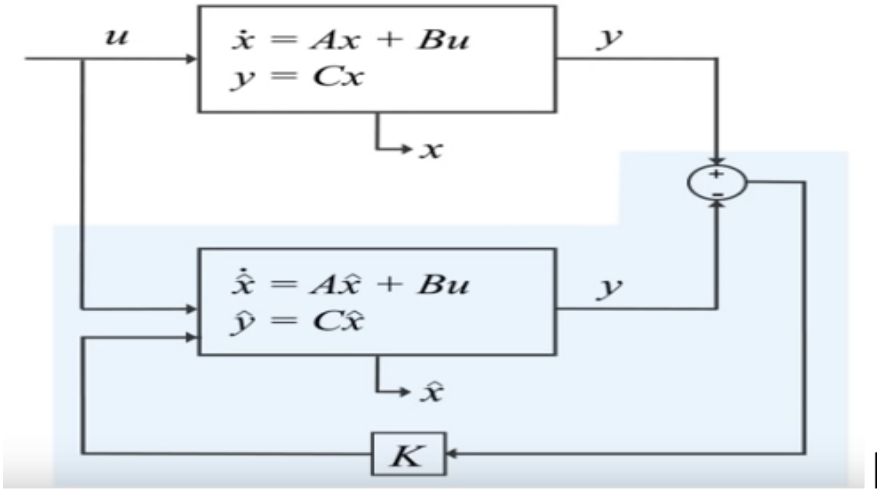

Figure 3.4. Real system and state observer [25].

The difference between x and $\hat{x}$ is called error and the difference between system $\dot{x}$ and observer $\dot{\hat{x}}$ is called observer error or error dynamic.

$$\dot{x} = Ax + Bu \qquad y = Cx$$

$$\dot{\hat{x}} = A\hat{x} + Bu + K(y - \hat{y}) \qquad \hat{y} = C\hat{x}$$

$$\mathrm{e}_{obs} = x - \hat{x}$$

$$y - \hat{y} = Cx - C\hat{x} = C(x - \hat{x}) = C\mathrm{e}_{obs}$$

Error dynamic function:

$$\dot{\mathrm{e}}_{obs} = \dot{x} - \dot{\hat{x}} = (A - KC)\mathrm{e}_{obs} \qquad (3.9)$$

The solution to the error dynamic function is exponential function. If $A - KC$ is less than 0 it means the error will decrease over the time and $\hat{x}$ will converge to x. Decay rate depends on to matrix A and because the A is not determined so there is no information about the disappearing of the error. To have more control over this equation the feedback controller is needed and it causes faster elimination of the error, and this causes the estimated state ($\hat{x}$) converges to true state (x) faster [25]. The optimal way of choosing the gain K is to perform to using Kalman filters.

In the moving object dynamic, velocity u_k is input, y_k is position and single state system x is object's position. To measure the state the matrix $C = 1$. It is important to know y as accurate as possible since it is important to object finish as close as possible to the finish line. There might be noisy measurement(r_k) that is earned by the sensors such as GPS, and the process noise such as friction or changing in the object velocity (p_k) are random variables. These random variables do not follow a specific pattern, with using probability theory, something about average properties is determined. Supposedly, r and p can be earned from normal distribution which is Gaussian distribution with 0 mean and covariance R ($r \sim N(0, R)$) . It means noise is close to 0 in measurement. Because the measurement is

noisy, the measurement does not quiet reflect the true position of the object. With having some information about the moving object, with running the input through the estimator which it will not be perfect ($\hat{x_k}$) because of process noise [25]. To solve the issue the Kalman filter is used to combine the measurement and the prediction to find the optimal estimate of the object position in the presence of process and measurement noises.

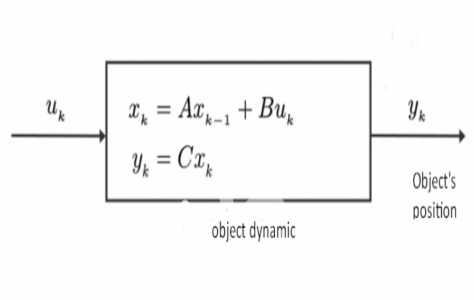

Figure 3.5. A system with x_K =[position] [25].

In this system matrix $C = 1$

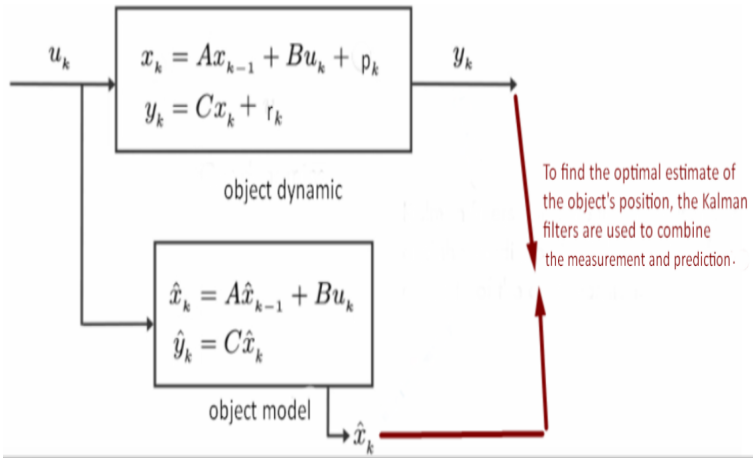

Figure 3.6. The Kalman filters principle.

The Kalman filter principle visually is shown as below:

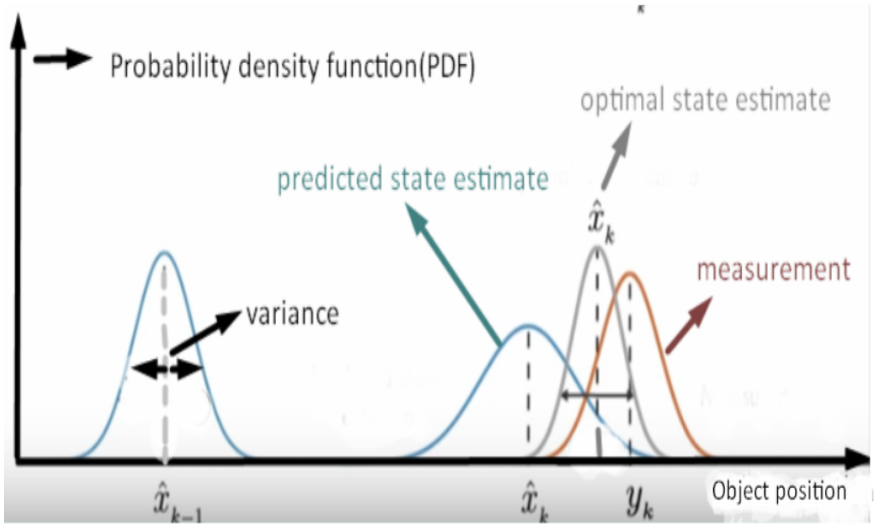

Figure 3.7. The Kalman filters principle in terms of probability density function and position.

To get the actual result in real world it is needed to implement the algorithm.

$$\hat{x}_{k+1} = A\hat{x}_k + Bu_k + K(y_k - C\hat{x}_k) \quad State\ observer\ of\ a\ deterministic\ system$$

$$\hat{x}_k = A\hat{x}_{k-1} + Bu_k + K_k(y_k - C(A\hat{x}_{k-1} + Bu_k)) \quad Kalman\ filter\ for\ a\ stochastic\ system$$

Where:

$\hat{x}_k^- = A\hat{x}_{k-1} + Bu_k$ is called prior estimate and predicts the current state and after substitution, the Kalman filter will be:

$$\hat{x}_k = \hat{x}_k^- + K_k(y_k - C\hat{x}_k^-)$$

It is called the posterior estimate and has two parts: $\hat{x}_k^-$ is called predict part and $K_k(y_k - C\hat{x}_k^-)$ is called update part.

So, the Kalman filter is two steps Process:

1) Prediction: $\begin{cases} \hat{x}_k^- = A\hat{x}_{k-1} + Bu_k \\ P_k^- = AP_{k-1}A^T + Q \end{cases}$

and

2) Update: $\begin{cases} \hat{x}_k = \hat{x}_k^- + K_k(y_k - C\hat{x}_k^-) \\ P_k = (I - K_kC)P_k^- \\ K_k = \frac{P_k^- C^T}{CP_k^- C^T + R} \end{cases}$

P_k^- is error covariance (uncertainty of estimate state), $\hat{x}_k^-$ is prior state estimate, K_k is the Kalman gain and Q is the covariance of the process noise.

In this work, the Kalman filter is used to estimate the position of a moving object in a 2D space from a series of noisy inputs based on the object past positions. it is a recursive filter that estimate the linear dynamic system state from a series of incomplete measurements. Here, the MATLAB version of the Kalman filter algorithm is implemented. The physic's motion laws is used by the Kalman filter for estimating the new states [26].

$$x = x_0 + v\Delta t$$

$$y = y_0 + v\Delta t$$

$$v_x = v_{x_0} + a_x\Delta t$$

$$v_y = v_{y_0} + a_y\Delta t$$

Where:

x and y are the object position.

v_x and v_y are the object velocity.

a_x and a_y are the object acceleration.

Hint: dt can be used instead of Δt too.

The state transition matrix A, is made by the coefficient values of x, y, v_x, v_y, a_x, and a_y.

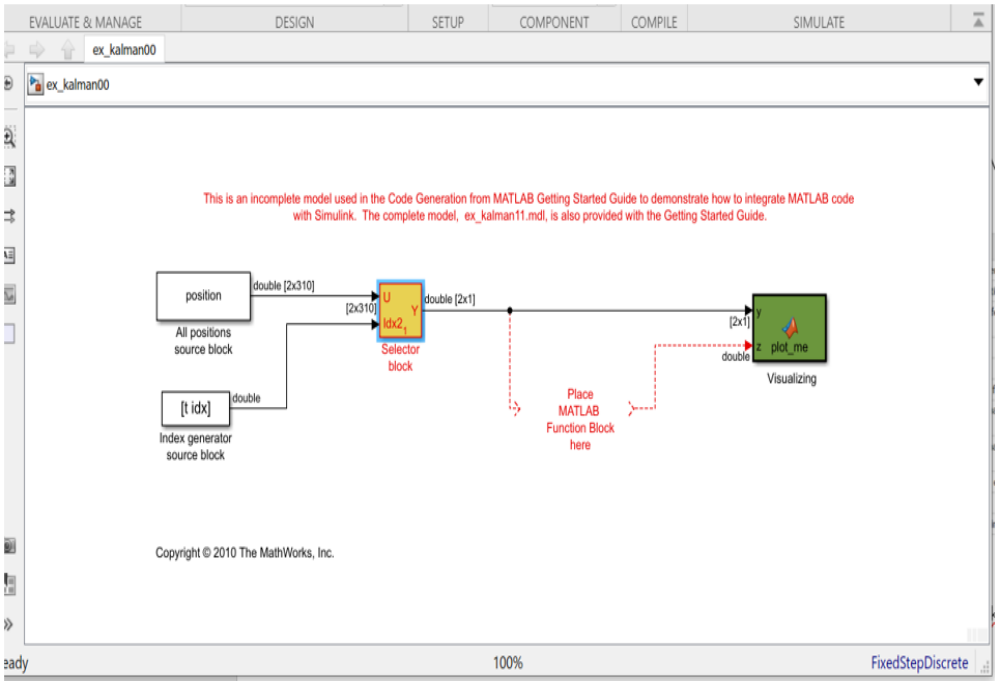

Figure 3.8. An incomplete Simulink model.

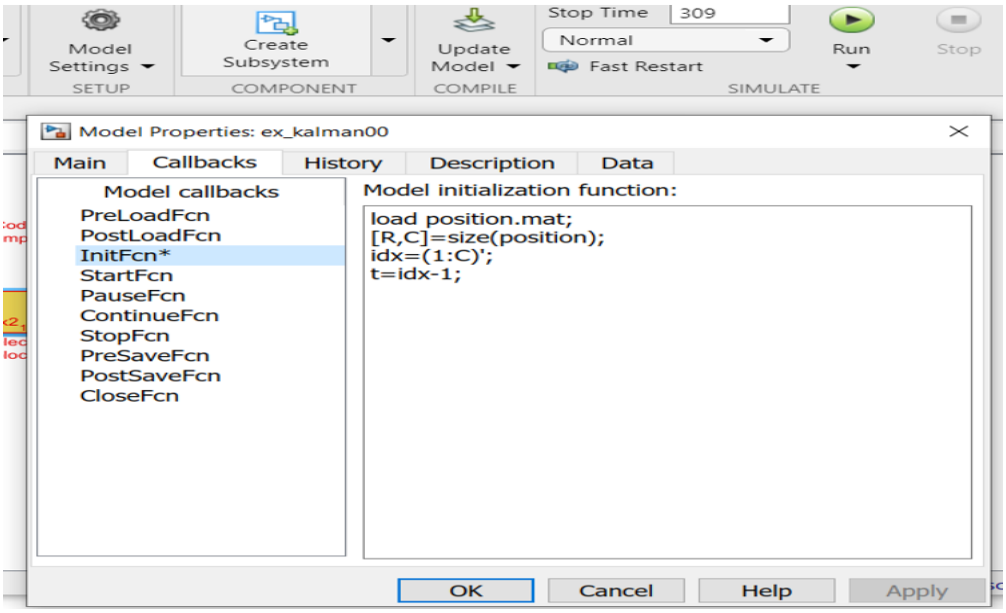

Figure 3.9. Call back function.

Callback function is used for loading mat file for position data and setting up the data that is used by generator block.

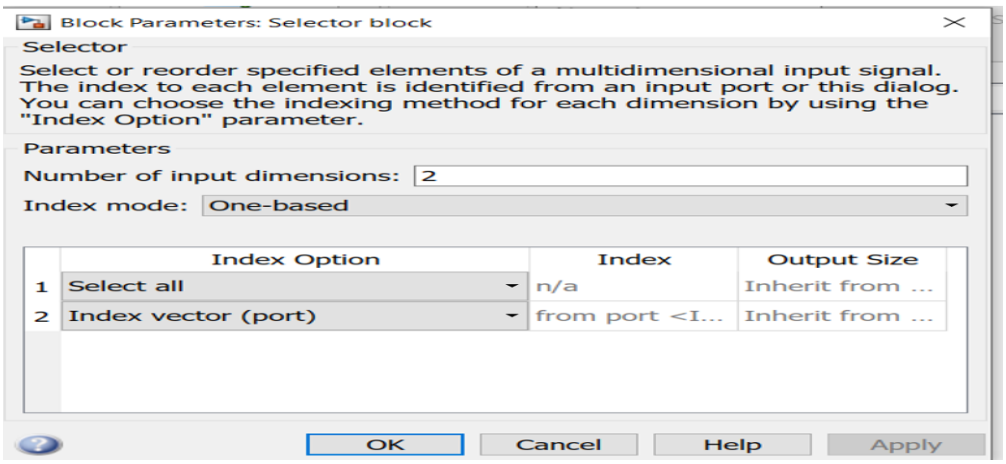

Figure 3.10. Selector block setting.

In selector block, input position matrix, index data increment and output at each time is determined.

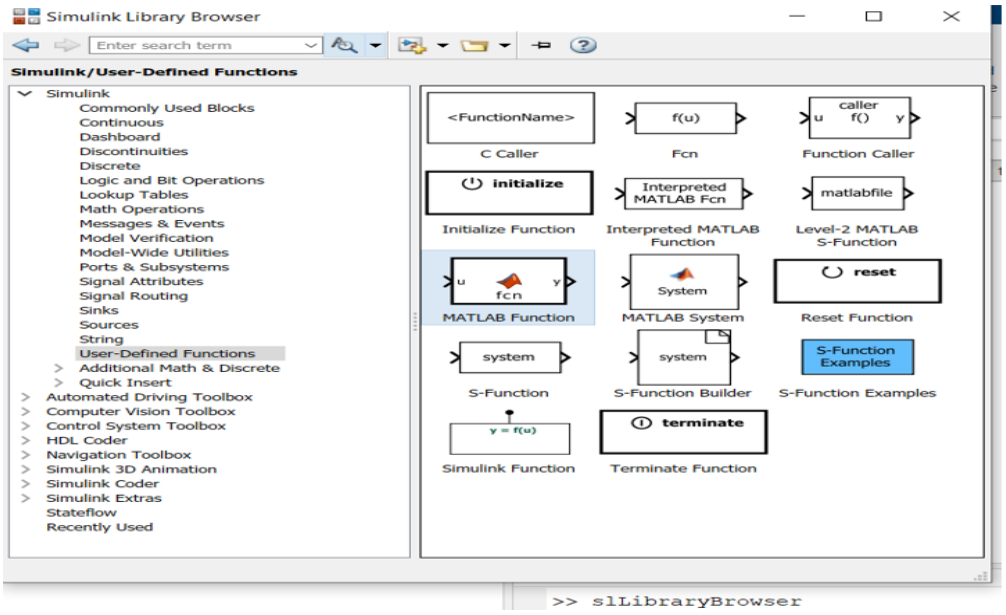

Figure 3.11. Adding MATLAB function block for Simulink library.

```
%ex_Kalman03.m file for fixed-size and variable-size input
function y = ex_reza03(z)
 %#codegen
dt=1;
A=[ 1 0 dt 0 0 0;...
   0 1 0 dt 0 0;...
   0 0 1 0 dt 0;...
   0 0 0 1 0 dt;...
   0 0 0 0 1 0 ;...
   0 0 0 0 0 1 ];
  H = [ 1 0 0 0 0 0; 0 1 0 0 0 0 ];
Q = eye(6);
R = 1000 * eye(2);
% Initial conditions
persistent x_est p_est
if isempty(x_est)
   x_est = zeros(6, 1);
   p_est = zeros(6, 6);
  end
% Pre-allocate output signal:
y=zeros(size(z));
for i=1:size(z,2)
   % Predicted state and covariance
   x_prd = A * x_est;
   p_prd = A * p_est * A' + Q;
    % Estimation
   S = H * p_prd' * H' + R;
   B = H * p_prd';
   klm_gain = (S \ B)';
    % Estimated state and covariance
   %x_est = x_prd + klm_gain * (z - H * x_prd);
   x_est = x_prd + klm_gain * (z(1:2,i) - H * x_prd);
   p_est = p_prd - klm_gain * H * p_prd;
    % Compute the estimated measurements
   %y = H * x_est;
   y(:,i) = H * x_est;
 end
end
```

Figure 3.12. The Kalman filter algorithm for fixed-size and variable-size input.

```
%call back function for fixed-size input

load position.mat;
[R,C]=size(position);
FRAME_SIZE=5;
idx=(1:FRAME_SIZE:C)';
LEN=length(idx);
t=(1:LEN)'-1;
==================================================
%call back function for variable-size input
load position.mat;
idx=[ 1 1 ;2 3 ;4 6 ;7 10 ;11 15 ;16 30 ;
     31 70 ;71 100 ;101 200 ;201 250 ;251 310];
LEN=length(idx);
t=(0:1:LEN-1)';
==========================================================
function plot_trajectory(z,y)
%#codegen
N=size(y,2);
coder.extrinsic('plot','title','xlabel','ylabel','axis','pause');
title('Trajectory of object [red] its Kalman estimate[blue]');
xlabel('position along x-axis');
ylabel('position along y-axis');
for i=1:N
    plot(z(1,i), z(2,i), 'rx-');
    plot(y(1,i), y(2,i), 'bo-');
    axis([-2.2, 2.2, -2.2, 2.2]);
    pause(0.01);
end
end
==================================
visualizing

function plot_me(y,z)
%#codegen

coder.extrinsic('figure','hold','plot_trajectory');
persistent h
if isempty(h)
  h=figure;
  hold;
end
```

Figure 3.13. Call back functions, plot trajectory function and visualizing.

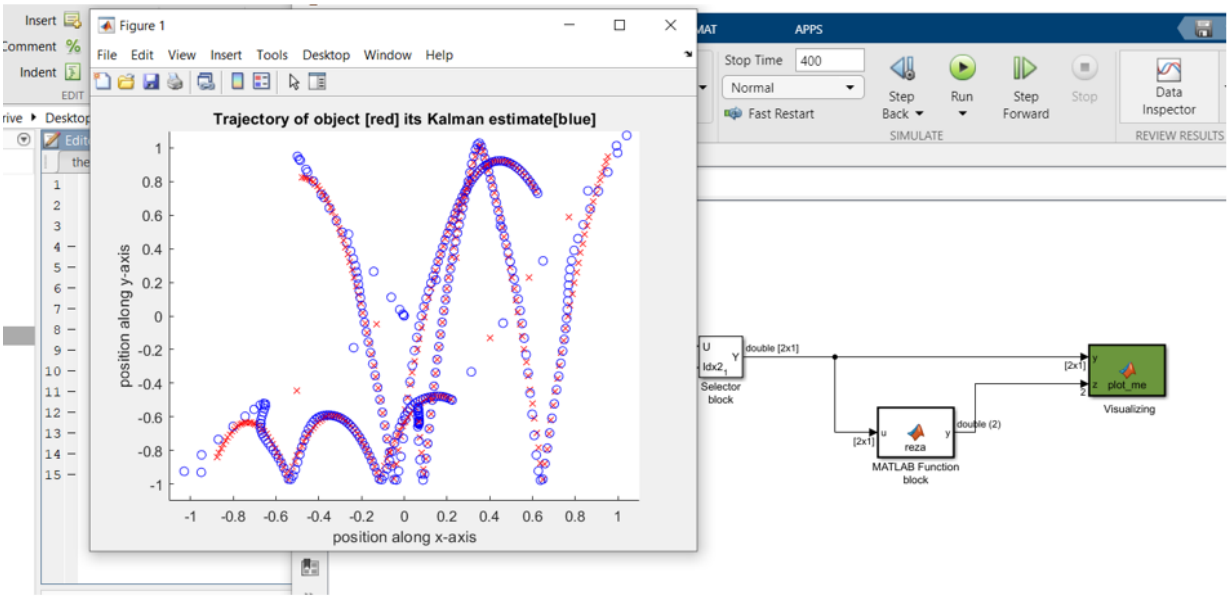

Figure 3.14. Object trajectory and its estimated position of fixed-size input in 400 stop time.

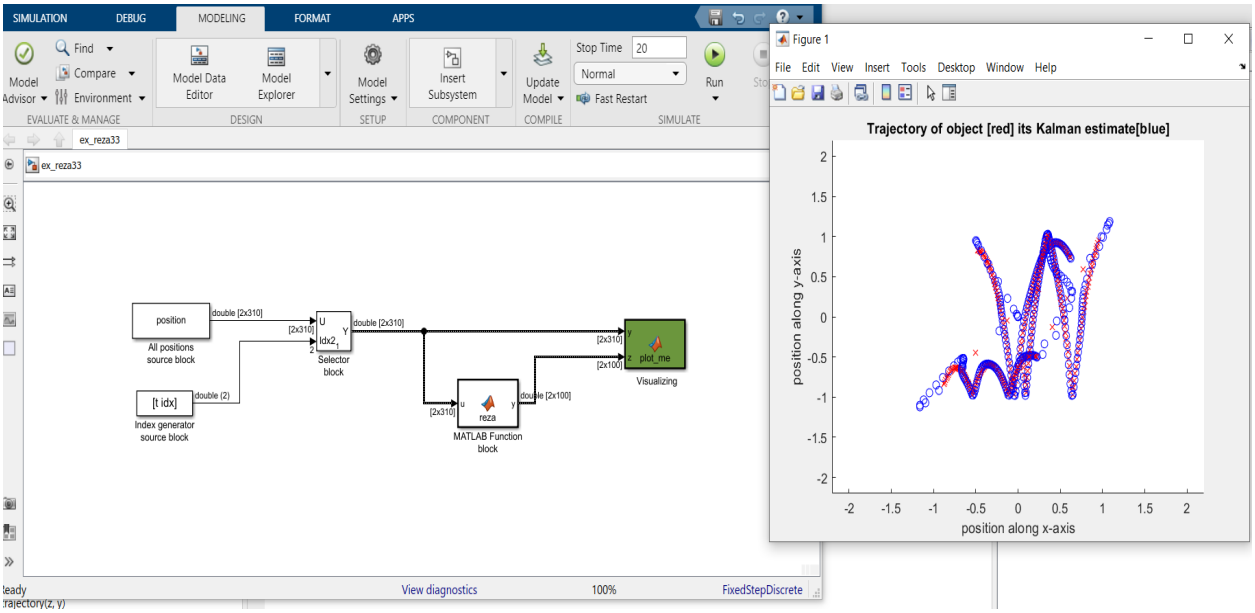

Figure 3.15. Object trajectory and its estimated position of variable-size input in 20 stop time.

When Simulink runs the model, the object trajectory is plotted in red and the Kalman filter estimated position is determined in blue.

For the model numbers (3.6a) and (3.6b), the state transition matrix A, with different coefficients is created and the Kalman filter algorithm will be:

```
% The Kalman filter algorithm for model x=vx.cos(θ) +Rx and y=vy.sin(θ) +Ry
function y = ex_reza04(z)
 %#codegen
theta=pi/6;
dRx=1;
dRy=1;
A=[ 1 0 sin(theta) 0 0 0;...
   0 1 0 cos(theta) 0 0;...
   0 0 1 0 dRx 0;...
   0 0 0 1 0 dRy;...
   0 0 0 0 1 0 ;...
   0 0 0 0 0 1 ];
H = [ 1 0 0 0 0 0; 0 1 0 0 0 0 ];
Q = eye(6);
R = 1000 * eye(2);
% Initial conditions
persistent x_est p_est
if isempty(x_est)
   x_est = zeros(6, 1);
   p_est = zeros(6, 6);
 end
% Pre-allocate output signal:
y=zeros(size(z));
for i=1:size(z,2)
   % Predicted state and covariance
   x_prd = A * x_est;
   p_prd = A * p_est * A' + Q;
    % Estimation
   S = H * p_prd' * H' + R;
   B = H * p_prd';
   klm_gain = (S \ B)';
    % Estimated state and covariance
   x_est = x_prd + klm_gain * (z(1:2,i) - H * x_prd);
   p_est = p_prd - klm_gain * H * p_prd;
      % Compute the estimated measurements
     y(:,i) = H * x_est;
end
end
```

Figure 3.16. The Kalman filter algorithm for (3.6a) and (3.6b) with different state transition matrix.

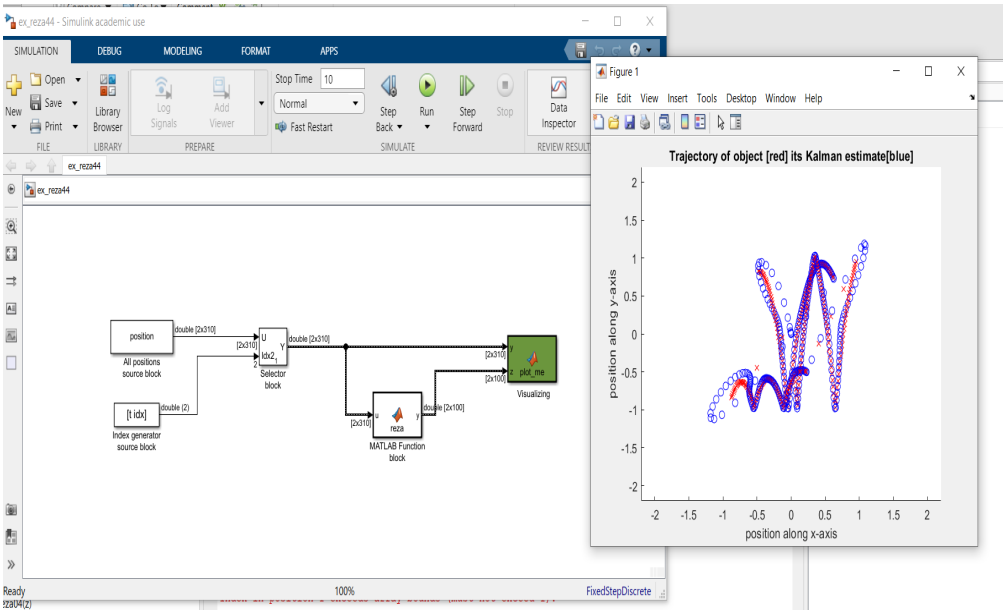

Figure 3.17. Object trajectory and its estimated position of variable-size input in 10 stop time.

The Kalman filter algorithm is implemented for fixed-size and variable-size algorithm.

3.6 Occupancy prediction model

Majority of collision in the drivable area is because of misbehavior of surrounding vehicle for numerous reasons. Predicting their future behavior is one of the important actions to choose the best strategy for preventing or reducing the collision. To do this, having other car information such corresponding position, velocity, and acceleration relative to AV help to make best decision and choose the safest path. With the given information about AV and relative surrounding car as an obstacle, following driving information (D_{c_n}) is obtained [22].

$$D_{c_n} = [p_{c_n,x}, p_{c_n,y}, v_{c_n,x}, v_{c_n,y}, a_{c_n,x}, a_{c_n,y}] \quad \text{for} \quad n \in 1, \ldots, \text{number of cars} \tag{3.10}$$

n is number of surrounding cars, but it can be used for any moving object or obstacle.

For example, if there is only one vehicle around the AV, the information for the vehicle will be:

$$n = 1 \qquad D_{c_1} = [p_{c_1,x}, p_{c_1,y}, v_{c_1,x}, v_{c_1,y}, a_{c_1,x}, a_{a_1,y}]$$

As previously mentioned:

$$D_{AV} = [p_{AV,x}, p_{AV,y}, v_{AV,x}, v_{AV,y}, a_{AV,x}, a_{AV,y}]$$

And the relative driving information between the ego car and surrounding vehicles is expressed as below [22]:

$$D_{r_n} = D_{c_n} - D_{AV} = [p_{c_n,x}, p_{c_n,y}, v_{r_n,x}, v_{r_n,y}, a_{r_n,x}, a_{r_n,y}] \qquad (3.11)$$

Where

D_{c_n}: is the set of surrounding vehicles driving information.

p_{c_n}: is the surrounding vehicle relative position.

v_{c_n}: is the surrounding vehicle velocity.

a_{c_n}: is the surrounding vehicle acceleration.

p_{AV}: is the ego car position.

v_{AV}: is the ego vehicle velocity.

a_{AV}: is the ego vehicle acceleration.

D_{r_n}: is the set of surrounding vehicle relative driving information.

v_{r_n}: is the surrounding vehicle relative velocity (v_{c_n} - v_{AV}).

a_{r_n}: is the surrounding vehicle relative acceleration (a_{c_n} - a_{AV}).

x and y are the fixed x-axis and y-axis with base AV, respectively [22].

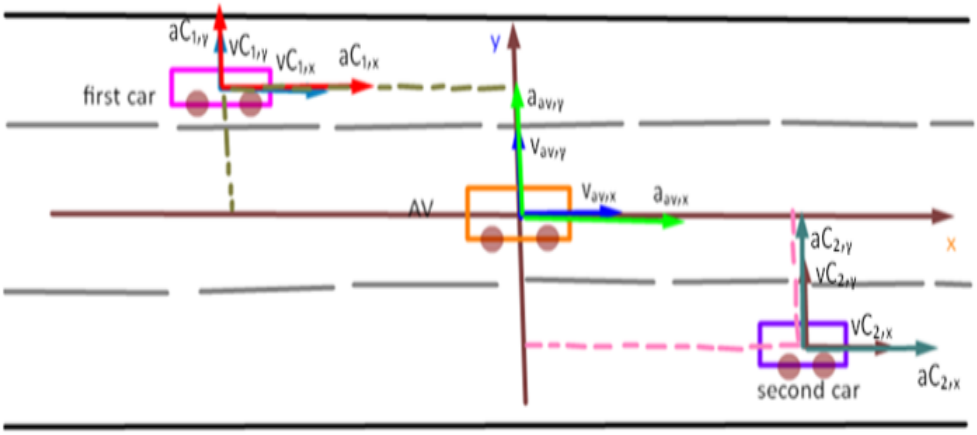

Figure 3.18. AV coordinate and relative information with surround vehicles [22].

In this prediction, which is the precise short-term prediction, is assumed the relative vehicle velocity and acceleration stay constant. It is one the best model for unexpected driving condition and the equation of the motion is expressed as follows:

$$\Delta d = \frac{1}{2} a_{r_n} \Delta t^2 + v_{r_n} \Delta t \qquad (3.12)$$

Where the Δd is the distance between AV and relative car during time Δt. If the initial time is considered zero, the above equation turns to [22]:

$$d = \frac{1}{2} a_{r_n} t^2 + v_{r_n} t$$

Any certain position which is the relative vehicle position (d_{c_n}), can be reached at time t_n which is called time-to-occupancy (TTO) as below:

$$T_n(d_{c_n}) = \frac{-v_{r_n} - \sqrt{v_{r_n}^2 - 2a_{r_n} d_{c_n}}}{a_{r_n}} \qquad (3.13)$$

Where T_n is time that AV arrives to a position that a relative car with the short distant d_{c_n} is located.

In case of AV's sudden velocity change which causes sudden acceleration/deceleration change, advance time to occupancy (ATTO) is offered. To calculate time to reach the position d_{c_n} with the considering the effect of acceleration change, below formula is offered.

$$AT_n(d_{c_n}) = \frac{d_{c_n}}{v_{r_n} + k\ a_{r_n}} \qquad (3.14)$$

Where AT_n is advanced time and k is given, and it is growth factor (growth constant) of acceleration that depends to several conditions such as road condition etc. [22].

3.7 Risk assessment

In AV technology, any unexpected and uncontrolled action in driving that create a dangerous situation for the driver and surrounding could be considered as source of risk. To prevent and mitigate the risk, a map is generated for surrounding obstacle and vehicles for the risk level evaluation for the car. To do this, some important information of the obstacle and surrounding vehicle such as position, velocity and acceleration is needed to assess the future position of AV. Predictive occupancy grid map determines the potential risk for surrounding vehicle based on their position, velocity, and acceleration. It helps to choose the best path with the minimum risk [22], [27].

With using ATTO, potential risk for surrounding vehicle is calculated by [22]:

$$R_n(d_{c_n}) = \frac{1}{AT_n(d_{c_n})} = \frac{v_{r_n} + ka_{r_n}}{d_{c_n}} \qquad (3.15)$$

4. BELIEF THEORY

4.1 Belief map generation

To generate an accurate occupancy grid, the Lidar output needs to be filtered before the unprocessed data with noisy measurement can be used for constructing occupancy grid. Once data filtering has been completed its 3D data needs to be converted to 2D by projection in the plane. Due to data complexity and sensor noise, there is still uncertainties in the grid map. To cancel the noise and reduce the uncertainties, binary occupancy grid is turned to probabilistic occupancy grid, means instead of cell i stores 0 and 1, they can be assigned a probability value between 0 and 1 depends on their occupancy position and the certainty that a corresponding cell is occupied. If the occupancy probability value of a cell is close to 1, it indicates the high certainty that the cell is occupied by an obstacle. Conversely, if the cell occupancy probability value is close to 0, it indicates the low certainty that the cell occupied which is considered free cell [6]. Belief map is the new occupancy grid representation to use this set of probabilities that is shown by the term bel. Each square or cell of occupancy grid is represented by m_{i} where i can be created from measurement γ and the vehicle location x. The belief over m_{i} is the probability that the cell m_{i} is occupied given the sensor measurement for that cell location. A belief is confident to classify a boundary (threshold) value as occupied, this helps to modify belief map to binary map. A belief map is made by [9]:

$$bel_t m_{\mathrm{i}} = p(m_{\mathrm{i}} \,|\, \gamma, x) \qquad (4.1)$$

m_{i} is map cell and γ is sensor measurement for m_{i}.

In below figure, value 0.94 is above boundary value, the corresponding cell considers occupied even the probability is not 1, and the value 0.05 is below boundary value, the corresponding cell considers free despite of the probability is not 0.

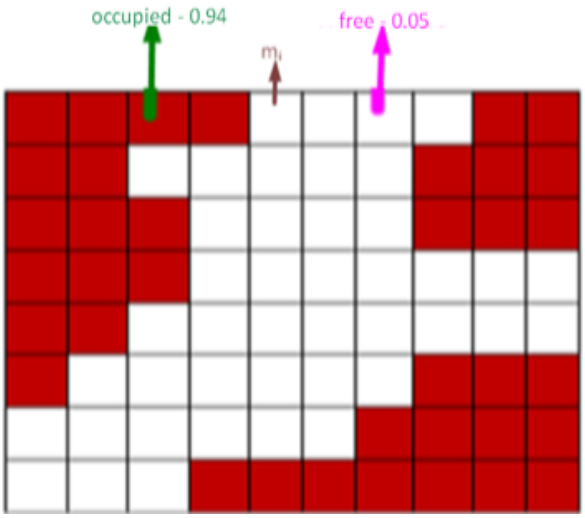

Figure 4.1. Probability occupancy map

4.2 Bayes' theorem and belief map accuracy

There is not enough precision with the belief for occupancy of a cell m_{i} with a single measurement, with having several measurements in different time from time 1 to t, more accurate belief of occupancy is gained. In this way, a combination of previous measurement from time 1 to t in a recursive way is used to make a more precise belief. The belief at time t over the map cell m_{i} in multiple time step, based on Bayes' theorem is define as below [9]:

$$bel_t(m_{\mathrm{i}}) = \; p(m_{\mathrm{i}} \,|\, (\gamma, x)_{1:t}) \qquad (4.2)$$

The probability that m_{i} is occupied, for given all measurement and vehicle position from time 1 to t.

To find the probability of a measurement at particular time t for map cell m_{i} , $p(\gamma_t \,|\, m_{\mathrm{i}})$ is used. It gives the probability of a particular measurement, while m_{i} is given.

So far, the belief for occupancy of a cell at each time step is calculated. With a recurrence relation, the belief map for occupancy of cell m_{i}, with using multiple time steps is earned as below:

$$bel_t(m_{\mathrm{i}}) \;=\; kp(\gamma_t \,|\, m_{\mathrm{i}}) bel_{t-1}(m_{\mathrm{i}}) \qquad (4.3)$$

Where: $p(\gamma_t \,|\, m_{\mathrm{i}})$ is current measurement at time t for cell m_{i} and $bel_{t-1}(m_{\mathrm{i}})$ is previous belief map for cell m_{i} and k is normalizing constant. To calculate belief map over m_{i} at time t-1, below equation is used:

$$bel_{t-1}(m_{\mathrm{i}}) \;=\; kp(\gamma_{t-1} \,|\, m_{\mathrm{i}}) bel_{t-2}(m_{\mathrm{i}})$$

$$\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots$$

$$bel_{t-n}(m_{\mathrm{i}}) \;=\; kp(\gamma_{t-n} \,|\, m_{\mathrm{i}}) bel_{t-n+1}(m_{\mathrm{i}}) \qquad (4.4)$$

4.3 Evidential autonomous model in belief theory

Because of obstacles in driving environment, the environment perception without driving aid will be uncertain and this brings the human driver confidence level down in driving commands. A belief function uses a reliable OGM and driving command decision as input and confidence level increase will be as output for it. Decision uncertainty varies corresponding to perception variation when the driving skills stays unchanged. Perception and localization are used on motion planning. There are several methods for computing the collision probability in case of uncertainty. OGM is the common way of environment representation. It is used to predict the collision risk and output for belief function for autonomous driving system. A model is needed to compute the confidence level related to the decision made by the autonomous driving system. The linear and angular velocity of the vehicle is used for decision making in belief theory and the decision and driving commands are expressed in term of them. By the degree of belief, the decision uncertainty is found in terms of degree of belief in the driving commands acceptability by using belief function theory. It makes the confidence level in the driving commands acceptability of the autonomous system [28].

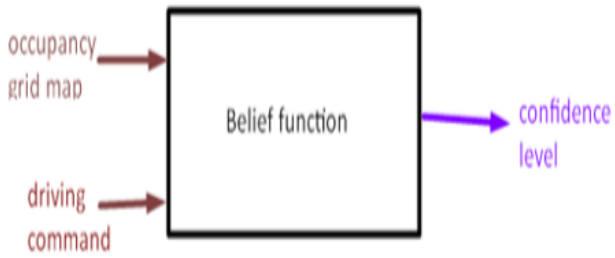

Figure 4.2. Input outpt diagram [28].

Driving commands, as a one of major input for belief function for having a reliable confidence level in an autonomous system is made by linear velocity v and angular velocity (ω), it means it is a function of v and ω. Sensors in an autonomous system is the way of collecting of all information and data from the surrounding, obstacles and their boundaries (threshold) which can be used if be represented in the form probabilistic in the discrete cells (Perrolaz et al. (2012)). Cells are the source of evidence of grid map and the decision is made based on occupancy information. If a cell is occupied, its probability is 1 (P(Oc =1)) and it is called the probabilistic occupancy [28].

The belief function theory (evidence theory) is a method that is used for reasoning uncertainty with using subjective probability or mass values. To calculate the probability, a finite set of possible interesting propositions must be given. Let consider the finite set $S = \{a,\ n\}$, where a is proposition of the admissible (occupied cell) state of the decision, and n is proposition of the non-admissible (free cell) state of decision. All possible proposition of interest is made by S is a finite set as follows [29], [28]:

$2^S = \{\ \phi,$ a , n, $S\ \}$ (2^S is called as a power set)

Where

ϕ: indicates no proposition.

S: indicates ignorance about the state of decision.

Function m is defined as below:

m: $2^S = \{\ \phi,$ a, n, $S\} \rightarrow [\ 0,\ 1]$

This function is called mass function, it assigns mass value to the proposition in the power set, and it satisfies the condition given below:

$$\sum_{X \subseteq S} m(X) = 1 \qquad (4.5)$$

Where $m(X)$ indicates the evidence quantity that the space is admissible (occupied), non-admissible (free), unknown and conflict.

Belief function in S is defined as below:

$$bel(Y) = \sum_{Y \subseteq X} m(Y) \qquad (4.6)$$

As it is mentioned before, the discrete cells of grid map are the evidence sources. In this model, the obstacles and road boundaries that are in the vehicle movement scope and the only cells that are in the candidate path is considered for evidence. The filter in occupancy grid map is used to separate evidence cells from others [28], [30].

The evidence discrete cells are divided in to three subsets:

1) Discrete cells in this area represent free space between vehicle and obstacles or road boundaries. In this area $P(Oc = 1)$ is very low.

2) Discrete cells in the area represent the obstacles position. In this area $P(Oc = 1)$ is very high.

3) Discrete cells is located at behind of obstacles and represents the ignorance. In this area there is no information of other obstacles. In this area $P(Oc = 1) = 0.5$.

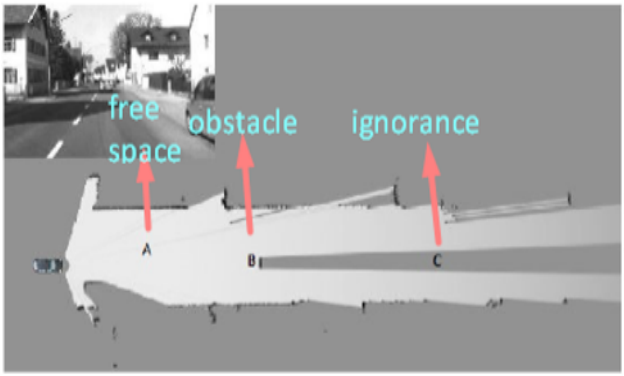

Figure 4.3. Area of evidence discrete cells [28].

4.4 Evidential occupancy grids

As it is mentioned in previous section, in evidential framework, a discernment frame $S = \{a,\ n\}$ with the power set $2^S = \{\ \phi$, a , n, $S\ \}$ is defined. where a represents admissible (occupied cell) state and n represents non-admissible (free cell) state of decision. It is used to model a specific problem. A mass function $m(X)$, also known as a basic belief assignment is defined to support cell state quantitatively. It creates four masses $[m(a)m(n)m(S)m(\phi)]$ [31].

The AV-centered evidential grids is generated from the sensor model which is two types:

Scan grid: is built from sensor data and Lidar points. It gives the information about the scanned cells state. Those cells that are hit by a Lidar ray are considered as occupied, the cells between the sensors and the occupied cells are considered free and the other cells are unknown. The mass value depends on the grid resolution and sensor performances [31].

Map grid: It is generated with fusing of new scan grid at each step of perception. A conflicting mass is provided by the fusion rule [31].

A mass function $[m(a)\, m(n)\, m(S)\, m(\phi)]$ will be dedicated to a cell when the map grid is processed. $m(S)$ represent the uncertainty value, $m(a)$ represent occupied mass, $m(n)$ represent free mass, and $m(\phi)$ represent the conflict resulting from a combination of $m(a)$ and $m(n)$.

For Example:

The mass distribution $[m(a)\, m(n)\, m(S)\, m(\phi)] = [0.8\ 0\ 0.2\ 0]$ states there is belief 0.8 Occupied cell. In this case, the rest of the masses are considered unknown [31], [32].

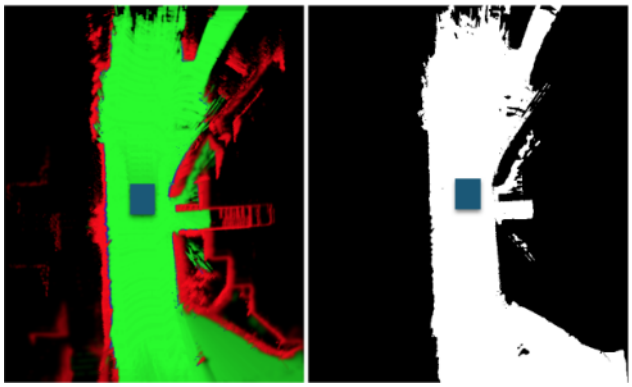

Figure 4.4. Evidential occupancy grid [31].

In above figure, left side is an evidential occupancy grid where :
The green color indicates the free cells (drivable area).
The red color indicates the occupied cells.
The blue color shows conflicting cells.
The black indicates unknown cells which is not explored.
The blue rectangle is vehicle position.

Right side grid is called binary grid is gained from evidential grid. For occupied cell the value 1 and for free cell the value 0 is dedicated. boundary(threshold) cells are considered as occupied state. Evidential grid is preferred because with the assigned mass to all subset of the domain in evidential grid, the uncertainty and conflict is determined [31].

Figure 4.5. A scene with corresponding evidential occupancy grid [33].

This is a good example of evidential occupancy grid where:

The yellow triangle represents car position (Lidar position).

The green area represents the free space.

The red area indicates occupied space.

The blue area is the conflicting cells.

The black color shows unknown cells (unexplored).

The degree of certainty is corresponding to color intensity.

4.5 Mass value application for decision admissibility

With the probabilistic occupancy of grid cells, the mass values are gained. Supposedly, cell 'i' has probabilistic occupancy $P(Oc = 1) = 0.6$, it can be said the cell 'i' is part of the target vehicle path at a distance 'd' from this vehicle.

If $P(Oc = 1) = x \rightarrow P(Oc = 0) = 1 - x$, then the probability of acceptable decision based on linear and angular velocity is 1 - x, because x is the probability of occupancy and the decision is made for rest of it, and it is represented by:

$P(\text{Decision}(v,\omega) \text{ is admissible}) = P(a) = 1 - x$. Applying break and obstacle motion are two important possibilities that effect the mass value assignment.

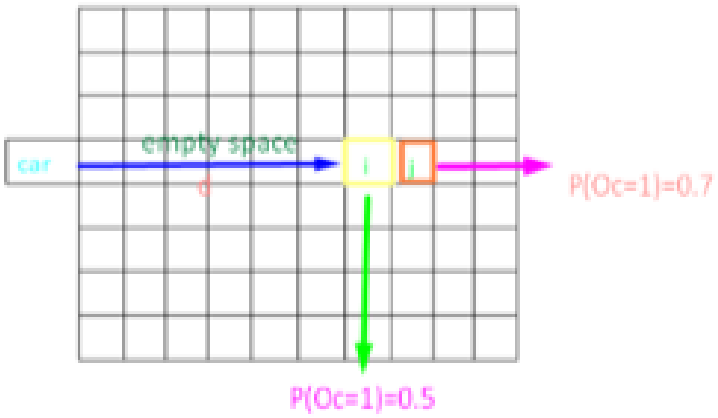

Figure 4.6. Sample occupancy grid

If an AV stops before collision it must apply these two equations respectively for linear and angular motion:

$$v^2 - v_0^2 = -2ad \qquad (4.7)$$

Where: v and v_0 are final and initial velocity respectively, a is acceleration (negative sign here is because the car stops and has deceleration) and d is distance between car and obstacle.

$$\omega^2 - \omega_0^2 = -2\alpha d \qquad (4.8)$$

Where ω and ω_0 are final and initial angular velocity respectively, α is angular acceleration. When a car stops before collision it means v and ω turn to zero and after simplifying two equations are gained as follows:

$$v_0 = \sqrt{2ad}$$

and

$$\omega_0 = \sqrt{2\alpha d}$$

If v_0 and ω_0 are replaced by v and ω, the following equations are obtained:

$$v = \sqrt{2ad}$$

and

$$\omega = \sqrt{2\alpha d}$$

So, both in equality in below must be satisfied till car be able stop before collision.

$$v \leq \sqrt{2da_{max}} \qquad (4.9)$$

and

$$\omega \leq \sqrt{2d\alpha_{max}} \qquad (4.10)$$

Road condition, tire condition and break condition and velocity etc. effects the car maximum deceleration for stopping. Lack of mentioned factors decrease the certainty of car halt before the collision. This uncertainty is expressed as probability of car stopping before collision. The Equation (4.9) and (4.10) do not guarantee the car stopping before collision. They only increase the probability of car halt before collision. This probability helps to calculate mass assignment function. Depends on satisfaction and dissatisfaction of decision based on v and ω of Equations (4.9) and (4.10) the mass value for proposition a for a particular cell i is expressed by:

If decision based on v and ω satisfies the Equations (4.9) and (4.10), in this case the probability of car stopping before collision is not zero and mass value for decision admissibility for a cell i is calculated by:

$$m_{\mathrm{i}}(a) = P(i \in A \text{ or } (i \in B \text{ and vehicle can stop before collision})) \quad \text{(satisfaction)}$$

With Equation (4.5), $m_{\mathrm{i}}(n)$ and $m_{\mathrm{i}}(S)$ are calculated. mass value for proposition $\emptyset$ is zero ($m_{\mathrm{i}}(\emptyset) = 0$).

If decision based on v and ω does not satisfy the Equations (4.9) and (4.10), in this case the probability of car stopping before collision is zero and mass value for decision admissibility for a cell i is calculated by:

$$m_{\mathrm{i}}(a) = P(i \in A) + P(i \in B).P(\text{vehicle can stop before collision}) \quad \text{(dissatisfaction)}$$

Same way $m_{\mathrm{i}}(n)$ and $m_{\mathrm{i}}(S)$ are calculated.

So far, the mass value for decision admissibility for a particular cell i is calculated. It can be applied to find confidence level [28].

4.6 Dempster-Shafer Theory in combination

In many fields, realizing the uncertainties helps to make the right decision. The events uncertainties are measured By Dempster-Shafer Theory (DST). Dempster-Shafer Theory, which is known as belief function theory or evidence theory, determines the foundation of decision-making. It is used to process the uncertain data in data fusion. With DST, an interval can be estimated instead of point estimation [34].

Recall: mass function is defined as below:

$m : 2^S = \rightarrow [\ 0,\ 1]$

Such that:

$$\sum_{X \subseteq S} m(X) = 1 \qquad and \qquad m(\phi) = 0$$

S is called frame of discernment (FOD) and belief function in S is defined as below:

$$bel(X) = \sum_{Y \subseteq X} m(Y)$$

The plausibility function is defined as:

$$pl(X) = \sum_{Y \cap X = \phi} m(Y) \qquad (4.11)$$

$$pl(X) = 1 - bel(X)$$

Since the mass value is for a single specific cell, with combining the mass values the confidence level is gained. The combination rules are used to fuse information from multiple sources [28].

Normalized Dempster's combination rules:

It relies on the degree of conflict for the normalization of combined mass values. Suppose m_1 and m_2 be two mass value functions on two propositions X and Y such that $X \cap Y \neq \emptyset$, the degree of conflict for these two functions is calculated by [28]:

$$r = \sum_{X \cap Y \neq \phi} m_1(X) m_2(Y) \qquad (4.12)$$

The degree of conflict is reliable if it is less than 1 and in this case two functions can be combined and combination for an admissible proposition Z such that $X \cap Y = Z \neq \emptyset$ is represented by following formula [28]:

$$(m_1 \bigoplus m_2)(Z) = \frac{r}{1-r} \quad (\text{for all } Z \neq \emptyset \text{ and } r < 1) \qquad (4.13)$$
$$= \frac{\sum m_1(X) m_2(Y)}{1 - \sum m_1(X) m_2(Y)}$$

where: $X \cap Y = Z$

For proposition $\emptyset$, it can be written $(m_1 \bigoplus m_2)(\emptyset) = 0$

The confidence level (C_l) for the decision (v, ω) is given by:

$$C_l(v, \omega) = bel(Z) = (m_i \bigoplus m_j)(Z) \qquad (4.14)$$

Where: i and j are two different discrete cells and confidence level is defined as belief function [28].

The belief function can be improved with modifying the basic probability assigned to mass function before data fusion. Let T be another set of propositions. The improved belief function is:

$$n(X) = \frac{1}{k} \qquad X \subseteq S \qquad (4.15)$$

Where: n is the propositions number with initial belief degree. Mass function for modified basic probability assignment will be:

$$m_I(X) = \frac{n(X) + m(X)}{1 + \frac{2^S - 1}{k}} \qquad (4.16)$$

Where m_I is improved mass function or modified basic probability assignment [34].

Example:

Suppose that the $S = \{a, b, c\}$ and basic probability assignments are:

$$m_1(a) = 0.95, \quad m_1(a, b) = 0.05$$

$$m_2(b) = 0.02, \quad m_2(c) = 0.98$$

Find the improved base belief function and modified mass function.

S has $2^3 = 8$ subsets, but 4 of them ({a}, {b} , {c}, {a, b}) have mass function values, they are called focal subset (element) in m_1 and m_2. So, $k = 4$ and $2^S = 2^3 = 8$.

The improved value of base belief function is: $n(X) = \frac{1}{4} = 0.25$

$$m_{I_1}(X) = \frac{n(X) + m_1(X)}{1 + \frac{2^S - 1}{k}} \Rightarrow m_{I_1}(a) = \frac{n(a) + m_1(a)}{1 + \frac{2^S - 1}{k}} = \frac{0.25 + 0.95}{1 + \frac{7}{4}} = 0.44$$

$$m_{I_1}(X) = \frac{n(X) + m_1(X)}{1 + \frac{2^S - 1}{k}} \Rightarrow m_{I_1}(a, b) = \frac{n(a, b) + m_1(a, b)}{1 + \frac{2^S - 1}{k}} = \frac{0.25 + 0.05}{1 + \frac{7}{4}} = 0.11$$

The basic probability assignment for those subsets are not given is zero and the modified values are:

$$m_{I_1}(b) = m_{I_1}(c) = m_{I_1}(a, c) = m_{I_1}(b, c) = m_{I_1}(a, b, c) = \frac{0.25}{1 + \frac{7}{4}} = 0.15$$

$$m_{I_2}(b) = \frac{n(b) + m_2(b)}{1 + \frac{2^S - 1}{k}} = \frac{0.25 + 0.02}{1 + \frac{7}{4}} = 0.098$$

$$m_{I_2}(c) = \frac{n(c) + m_2(c)}{1 + \frac{2^S - 1}{k}} = \frac{0.25 + 0.98}{1 + \frac{7}{4}} = 0.45$$

$$m_{I_2}(a) = m_{I_2}(a,b) = m_{I_2}(a,c) = m_{I_2}(b,c) = m_{I_2}(a,b,c) = \frac{0.25}{1+\frac{7}{4}} = 0.15$$

In the previous sections, the importance of the occupancy grid map to avoid collisions has been stated. In this section, the importance and application of the belief function in creating a complete, efficient and useful grid map is represented. Environment perception is significant for an AV in order to have less-error navigation. Occupancy grid is the main tool, that manage to have a deep and precise perception. In complex environment perception in terms of geometrical complexity, occupancy grid build a framework to overcome the complex condition. Lidar system characterize the vehicle dynamic environment. Map estimation is done by Bayesian rule and evidential framework according to Dempster-Shafer theory in belief functions. The accumulative information is filtered by the map grid. Also, moving objects are detected by map grid [35].

5. CONCLUSIONS

In this thesis, levels of autonomy and the information gaining tools from surrounding is mentioned. The Occupancy Grid framework is studied in probabilistic and stereo vision approach. It is stated how occupancy probability of a cell can be estimated by binary Bayes rule. It is mentioned that how a local grid map is generated by stereo camera in AV. The results from its application to ego vehicle mapping and navigation tasks in unknown and unstructured environments is presented. Inflation of an object with its algorithm implementation is discussed. Predictive occupancy is used in collision avoidance system to predict obstacle position. The Kalman filter and its application on tracking an object is investigated. Then, the belief theory and its application which significantly improves the map building process for intelligent vehicles environment perception and grid map estimation is discussed. Dempster-Shafer Theory is utilized to combine the mass values and improvement of base belief function is discussed.

6. FUTURE WORK

Because of the vastness of AV topic, there is a lot of work to be done in the future. Developing the machine learning algorithm that has smaller time complexity and less computational cost. Applying more number of sensors to get more information of surrounding. Updating the stereo camera to gain the depth information of surrounding like the Lidar data. Utilizing some types of object tracker that be able to track the obstacle in vertical and horizontal movement with constant and variable velocity. The Investigated Grid Map (IGM) is working for small area of surrounding, expanding it for longer distance is the future approach.

REFERENCES

[1] A. P. Dhongade and M. A. Khandekar, "GPS and IMU integration on an autonomous vehicle using kalman filter (labview tool)," *2019 International Conference on Intelligent Computing and Control Systems (ICCS)*, pp. 1122–1125, 2019.

[2] F. M. Favarò and N. Nader, "Examining accident reports involving autonomous vehicles in California," *PLoS One*, vol. 126, pp. 151–168, 1 2017. DOI: 10.1371/journal.pone.0184952.

[3] H. Veeramachaneni, *Sensor technologies on one-thenth scale of an autonomous race car*, Masters of Science in Electrical and Computer Engineering IUPUI, Dec. 2020.

[4] S. Grigorescu, B. Trasnea, T. Cocias, and G. Macesanu, "A survey of deep learning techniques for autonomous driving," *Journal of Field Robotics*, vol. 37, Nov. 2019. DOI: 10.1002/rob.21918.

[5] P. Marín, J. Beltrán, A. Hussein, B. Musleh, D. Martín Gómez, A. de la Escalera, and J. Armingol, "Stereo vision-based local occupancy grid map for autonomous navigation in ros," Mar. 2016. DOI: 10.5220/0005787007010706.

[6] (2021). Occupancy grids. Section 2.3, [Online]. Available: https://www.mathworks.com/help/robotics/ug/occupancy-grids.html. /(Last Accessed: 05/24/2021).

[7] (2020). Create occupancy grid with binary values. Section 2.3, [Online]. Available: https://www.mathworks.com/help/robotics/ref/binaryoccupancymap.html. /(Last Accessed: 10/12/2020).

[8] (2021). Create occupancy map with probabilistic values. Section 2.5, [Online]. Available: https://www.mathworks.com/help/nav/ref/occupancymap.html. /(Last Accessed: 04/20/2021).

[9] Coursera, *Occupancy grids*, 2021. [Online]. Available: https://www.coursera.org/lecture/motion-planning-self-driving-cars/lesson-1-occupancy-grids-oJcwU, /(Last Accessed: 05/10/2021).

[10] (2019). Grid maps. Section 2.6, [Online]. Available: http://ais.informatik.uni-freiburg.de/teaching/ws12/mapping/pdf/slam11-gridmaps-4.pdf. /(Last Accessed: 02/25/2021).

[11] G. Wagner, *Smart collision avoidance for autonomous vehicles*, Masters of science Department of Space, Earth and Environment, Jun. 2018.

[12] K. Y. Kok and P. Rajendran, "A review on stereo vision algorithm: Challenges and solutions," vol. 13, pp. 134–150, Nov. 2019. DOI: 10.37936/ecti-cit.2019132.194324.

[13] A. Krishnan and J. Kollipara, "Intelligent indoor mobile robot navigation using stereo vision," *Signal & Image Processing : An International Journal*, vol. 5, Sep. 2014. DOI: 10.5121/sipij.2014.5405.

[14] M. Al-Azawi, D. Al-Rawy, S. Al-Jobory, A. Prof, M. Student, and D. Zaghar, "Stereo vision for 3d measurement in robot systems," *ResearchGate*, p. 13, Mar. 2019.

[15] B. Musleh, D. Martín, J. M. Armingol, and A. De la Escalera, "Pose self-calibration of stereo vision systems for autonomous vehicle applications," *Sensors*, vol. 16, no. 9, 2016, ISSN: 1424-8220. DOI: 10.3390/s16091492. [Online]. Available: https://www.mdpi.com/1424-8220/16/9/1492.

[16] M. Gosta and M. Grgic, "Accomplishments and challenges of computer stereo vision," *Proceedings Elmar - International Symposium Electronics in Marine*, pp. 57–64, Oct. 2010.

[17] (2021). Occupancy map. Section 2.8, [Online]. Available: https://www.mathworks.com/help/nav/ref/occupancymap.html#mw_16b90056-a9d4-4dcb-9ef4-f7c81acb66b8. /(Last Accessed: 03/28/2021).

[18] (2021). Occupancy map 3d. Section 2.8, [Online]. Available: https://www.mathworks.com/help/nav/ref/occupancymap3d.html. /(Last Accessed: 06/20/2021).

[19] (2021). Occupancy grids. Section 2.9, [Online]. Available: https://www.mathworks.com/help/robotics/ug/occupancy-grids.html. /(Last Accessed: 06/28/2021).

[20] (2021). The next wave in safety technology: Collision avoidance systems. Section 2.8, [Online]. Available: https://www.samsara.com/guides/collision-avoidance-system. /(Last Accessed: 05/10/2021).

[21] OrCAD, *How do collision avoidance systems work?* 2021. [Online]. Available: https://www.orcad.com/cn/node/6581, /(Last Accessed: 05/25/2021).

[22] K. Lee and D. Kum, *Collision avoidance/mitigation system: Motion planning of autonomous vehicle via predictive occupancy map*, Apr. 11, 2019. [Online]. Available: https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=8698763.

[23] D. Shen, Y. Chen, L. Li, and S. Chien, "Collision-free path planning for automated vehicles risk assessment via predictive occupancy map," in *2020 IEEE Intelligent Vehicles Symposium (IV)*, 2020, pp. 985–991. DOI: 10.1109/IV47402.2020.9304720.

[24] R. Hayashi, J. Isogai, P. Raksincharoensak, and M. Nagai, "Autonomous collision avoidance system bycombined control of steering and braking usinggeometrically optimised vehicular trajectory," *Vehicle System Dynamics*, vol. 126, pp. 151–168, 2012. DOI: https://doi.org/10.1080/00423114.2012.672748.

[25] M. Ulosoy, *Title of youtube video*, https://www.youtube.com/watch?v=mwn8xhgNpFY, Youtube Comment, In comment section, Last Accessed: 06/20/2021, 2018.

[26] (2021). Track object using MATLAB code. Section 3.0, [Online]. Available: https://www.mathworks.com/help/simulink/ug/tutorial-integrating-matlab-code-with-a-simulink-model-for-tracking-a-moving-object.html#bslqac6-3. /(Last Accessed: 07/01/2021).

[27] S. Lefèvre, D. Vasquez, and C. Laugier, *A survey on motion prediction and risk assessment forintelligent vehicles*, Aug. 1, 2014. [Online]. Available: https://hal.inria.fr/hal-01053736/document, /(Last Accessed: 05/25/2021).

[28] S. C. Jugade and A. C. Victorino, "Grid based estimation of decision uncertainty of autonomous driving systems using belief function theory," 2018. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2405896318307663, /(Last Accessed: 03/15/2021).

[29] H. Laghmara, M.-T. Boudali, T. Laurain, J. Ledy, R. Orjuela, J.-P. Lauffenburger, and M. Basset, "Obstacle avoidance, path planning and control for autonomous vehicles," in *2019 IEEE Intelligent Vehicles Symposium (IV)*, Jun. 2019, pp. 529–534. DOI: 10.1109/IVS.2019.8814173.

[30] S. Wirges, C. Stiller, and F. Hartenbach, "Evidential occupancy grid map augmentation using deep learning," in *2018 IEEE Intelligent Vehicles Symposium (IV)*, 2018, pp. 668–673. DOI: 10.1109/IVS.2018.8500635.

[31] H. Mouhagir, R. Talj, V. Cherfaoui, F. Aioun, and F. Guillemard, "Trajectory planning for autonomous vehicle in uncertain environment using evidential grid," in *20th International Federation of Automatic Control World Congress (IFAC WC 2017)*, vol. 50, Toulouse, France, Jul. 2017, pp. 12 545–12 550. [Online]. Available: https://hal.archives-ouvertes.fr/hal-01556594, /(Last Accessed: 04/28/2021).

[32] J. Moras, V. Cherfaoui, and P. Bonnifait, "Credibilist occupancy grids for vehicle perception in dynamic environments," in *2011 IEEE International Conference on Robotics and Automation*, 2011, pp. 84–89. DOI: 10.1109/ICRA.2011.5980298.

[33] H. Mouhagir, V. Cherfaoui, R. Talj, F. Aioun, and F. Guillemard, "Using Evidential Occupancy Grid for Vehicle Trajectory Planning Under Uncertainty with Tentacles," in *20th IEEE International Conference on Intelligent Transportation (ITSC 2017)*, Yokohama, Japan, Oct. 2017, pp. 1–7. [Online]. Available: https://hal.archives-ouvertes.fr/hal-01576974.

[34] S. Ni, Y. Lei, and Y. Tang, "Improved base belief function-based conflict data fusion approach considering belief entropy in the evidence theory," *Entropy*, vol. 22, no. 8, 2020, ISSN: 1099-4300. DOI: 10.3390/e22080801. [Online]. Available: https://www.mdpi.com/1099-4300/22/8/801, /(Last Accessed: 04/17/2021).

[35] J. Dezert, J. Moras, and B. Pannetier, "Environment perception using grid occupancy estimation with belief functions," in *2015 18th International Conference on Information Fusion (Fusion)*, 2015, pp. 1070–1077.