Fusion de données multi capteurs pour la détection et le suivi d'objets mobiles à partir d'un véhicule autonome

Qadeer Baig

UNIVERSITÉ DE GRENOBLE

THÈSE

Pour obtenir le grade de

DOCTEUR DE L'UNIVERSITÉ DE GRENOBLE

Spécialité : **Mathématique, Informatique (Robotique)**

Arrêté ministériel : 7 août 2006

Présentée par

Qadeer Baig

Thèse dirigée par **Thierry FRAICHARD** et
codirigée par **Olivier AYCARD**

préparée au sein du **Laboratoire d'Informatique de Grenoble**
dans **Mathématiques, Sciences et Technologies de l'Information, Informatique**

Fusion de données multi capteurs pour la détection et le suivi d'objets mobiles à partir d'un véhicule autonome

Thèse soutenue publiquement le **29 Février 2012**,
devant le jury composé de :

Mr Laurent TRASSOUDAINE
Professeur, Université Blaise Pascal, Rapporteur
Mme Véronique Berge CHERFAOUI
MCF HDR, Université de Technologie de Compiègne, Rapporteur
Mr Didier AUBERT
Directeur de Recherche, IFSTTAR Paris, Président
Mme Michèle ROMBAUT
Professeur, UJF, Examinatrice
Mr Thierry FRAICHARD
CR HDR, INRIA Grenoble, Directeur de thèse
Mr Olivier AYCARD
MCF HDR, Université de Grenoble, Co Directeur de thèse

Multisensor Data Fusion for Detection and Tracking of Moving Objects From a Dynamic Autonomous Vehicle

To:

Ammi

Abu

Tiayyba

Safiyy

Abstract

Perception is one of important steps for the functioning of an autonomous vehicle or even for a vehicle providing only driver assistance functions. Vehicle observes the external world using its sensors and builds an internal model of the outer environment configuration. It keeps on updating this internal model using latest sensor data. In this setting perception can be divided into two sub parts: first part, called *SLAM* (Simultaneous Localization And Mapping), is concerned with building an online map of the external environment and localizing the host vehicle in this map, and second part deals with finding moving objects in the environment and tracking them over time and is called *DATMO* (Detection And Tracking of Moving Objects). Using high resolution and accurate laser scanners successful efforts have been made by many researchers to solve these problems. However, with low resolution or noisy laser scanners solving these problems, especially *DATMO* , is still a challenge and there are either many false alarms, miss detections or both. In this thesis we propose that by using vision sensor (mono or stereo) along with laser sensor and by developing an effective fusion scheme on an appropriate level, these problems can be greatly reduced.

The main contribution of this research is concerned with the identification of three fusion levels and development of fusion techniques for each level for *SLAM* and *DATMO* based perception architecture of autonomous vehicles. Depending on the amount of preprocessing required before fusion for each level, we call them *low level*, *object detection level* and *track level* fusion. For low level we propose to use grid based fusion technique and by giving appropriate weights (depending on the sensor properties) to each grid for each sensor a fused grid can be obtained giving better view of the external environment in some sense. For object detection level fusion, lists of objects detected for each sensor are fused to get a list of fused objects where fused objects have more information then their previous versions. We use a Bayesian fusion technique for this level. Track level fusion requires to track moving objects for each sensor separately and then do a fusion between tracks to get fused tracks. Fusion at this level helps remove false tracks.

Second contribution of this research is the development of a fast technique of finding road borders from noisy laser data and then using these border information to remove false moving objects. Usually we have observed that many false moving objects appear near the road borders due to sensor noise. If they are not filtered out then they result into many false tracks close to vehicle making vehicle to apply breaks or to issue warning messages to the driver falsely.

Third contribution is the development of a complete perception solution for lidar and stereo vision sensors and its integration on a real vehicle demonstrator used for a European Union project (INTERSAFE-2[1]). This project is concerned with the safety at intersections and aims at the reduction of injury and fatal accidents there. In this project we worked in collaboration with Volkswagen, Technical university of Cluj-Napoca Romania and INRIA Paris to provide a complete perception and risk assessment solution for Volkswagen demonstrator vehicle.

Key Words:

Robotics, Multisensor data fusion, Fusion levels, Environment perception, SLAM, DATMO, Laser scanner, Stereo vision, Occupancy Grids, Road border detection.

[1] `http://www.intersafe-2.eu/public/`

Résumé

La perception est un point clé pour le fonctionnement d'un véhicule autonome ou même pour un véhicule fournissant des fonctions d'assistance. Un véhicule observe le monde externe à l'aide de capteurs et construit un modèle interne de l'environnement extérieur. Il met à jour en continu ce modèle de l'environnement en utilisant les dernières données des capteurs. Dans ce cadre, la perception peut être divisée en deux étapes : la première partie, appelée *SLAM* (Simultaneous Localization And Mapping) s'intéresse à la construction d'une carte de l'environnement extérieur et à la localisation du véhicule hôte dans cette carte, et deuxième partie traite de la détection et du suivi des objets mobiles dans l'environnement (*DATMO* Detection And Tracking of Moving Objects). En utilisant des capteurs laser de grande précision, des résultats importants ont été obtenus par les chercheurs. Cependant, avec des capteurs laser de faible résolution et des données bruitées, le problème est toujours ouvert, en particulier le problème du *DATMO*. Dans cette thèse nous proposons d'utiliser la vision (mono ou stéréo) couplée à un capteur laser pour résoudre ce problème.

La première contribution de cette thèse porte sur l'identification et le développement de trois niveaux de fusion. En fonction du niveau de traitement de l'information capteur avant le processus de fusion, nous les appelons "fusion bas niveau", "fusion au niveau de la détection" et "fusion au niveau du suivi". Pour la fusion bas niveau, nous avons utilisé les grilles d'occupations. Pour la fusion au niveau de la détection, les objets détectés par chaque capteur sont fusionnés pour avoir une liste d'objets fusionnés. La fusion au niveau du suivi requiert le suivi des objets pour chaque capteur et ensuite on réalise la fusion entre les listes d'objets suivis.

La deuxième contribution de cette thèse est le développement d'une technique rapide pour trouver les bords de route à partir des données du laser et en utilisant cette information nous supprimons de nombreuses fausses alarmes. Nous avons en effet observé que beaucoup de fausses alarmes apparaissent sur le bord de la route.

La troisième contribution de cette thèse est le développement d'une solution complète

pour la perception avec un capteur laser et des caméras stéréo-vision et son intégration sur un démonstrateur du projet européen (INTERSAFE-2[2]). Ce projet s'intéresse à la sécurité aux intersections et vise à y réduire les blessures et les accidents mortels. Dans ce projet, nous avons travaillé en collaboration avec Volkswagen, l'Université Technique de Cluj-Napoca, en Roumanie et l'INRIA Paris pour fournir une solution complète de perception et d'évaluation des risques pour le démonstrateur de Volkswagen.

Mots Clés:

Robotique, Fusion de donnée multi capteurs, Niveaux de fusion, Perception de l'environnement, SLAM, DATMO, Capteur laser, Stéréovision, Grilles d'occupations, Detection des bords de route

[2] http://www.intersafe-2.eu/public/

Acknowledgements

I am highly thankful to my directors of thesis Thierry Fraichard and Olivier Aycard for providing me an opportunity to work with them and for all their help during this time. I am especially thankful to Olivier Aycard for being ever ready to help me cope with technical and even administrative problems. Without his help this work would have been an impossible task for me.

I am thankful to my family for their prayers and moral support especially my wife and my son for being a source of motivation even in difficult times.

My special thanks to Higher Education Commission of Pakistan for supporting this work financially even when Pakistan was passing through financial problems.

This work was also partially supported by research project INTERSAFE-2. INTERSAFE-2 is part of the 7th Framework Programme, funded by the European Commission. The partners of INTERSAFE-2 thank the European Commission for supporting the work of this project.

Contents

CHAPTER 1

Introduction

Human have always longed and strove for getting more and more information about the external world with an objective of making their existence safer and more comfortable. Primarily they have used their five senses for this purpose. Looking on other living things it can be easily identified that nature has equipped them with senses which may differ in number, quality and type depending on the type of environment where they need to keep their survival. The fact that most of the living things have more than one senses simply implies that only one sense is not sufficient to get information about the environment necessary for survival. For example, assessing the quality of an edible substance may not be possible using only the sense of vision; the combination of sight, touch, smell, and taste is far more effective. Similarly, when vision is limited by structures or other obstructions in the way, the sense of hearing can provide advanced warning of impending dangers (Hall and Llinas, 2001). All living things are capable of combining information from all of their senses, without any effort, in the most informative way, enabling them to timely react to any changes in the environment.

Inspired from this natural observation, modern man has recognized that a true autonomous machine or robot must have multiple sensors to get information from the environment where it will work to achieve some goal. It is because of this fact that we see even semi autonomous robots being equipped with different sensors like lidars, vision sensors, radars etc. Although combining information from different senses is quite natural and an effortless process in living things, however mimicking the same capability in human made machines is an extremely challenging task. The realization of this challenge is becoming more and more viable due to the emergence of new sensors, advanced processing techniques and algorithms, and improved processing hardware. Recent advances in computing and sensing have provided the capability to emulate, in hardware and software, the natural data fusion capabilities of humans and animals.

Autonomous vehicle[1] driving is a hot field of research these days. Based on modern sensors and processing hardware researchers are developing techniques that will enable the intelligent vehicles to see the things around them on the roads and assess their behavior, an important prerequisite for their responsible autonomous control on the roads. However there are, as yet, many challenges to overcome in this field, especially related to extracting and combining useful information from sensors' data to understand external world.

The objective of our research in this dissertation has been to develop fusion techniques for multiple sensors (laser and vision (mono and stereo)) to improve perception capabilities of an autonomous vehicle. This research has resulted in the development of fusion strategies at different levels of abstraction for these sensors. The details of these techniques are given in rest of the chapters of this thesis whereas this chapter develops the context of our research.

In this chapter we first give a brief history of intelligent vehicles and then explain what intelligent vehicle perception is. We also introduce some concepts about sensors and briefly describe multi sensor data fusion. We close this chapter with a list of our contributions and thesis layout.

1.1 Brief history of intelligent vehicles

The types of robots that we want to see on roads or highways are autonomous vehicles, also called driver-less cars; driving people or goods from one point to the other without human intervention, sharing the roads with other driver-less or human driven cars and following the traffic rules. In the following we mention some earlier attempts to build vehicles having some degree of autonomy along with sensory instruments used by them to observe the world.

In retrospect the very first vehicle built that will attain some degree of autonomy was *Stanford Cart* [2] in 1961 with a research aim of controlling a moon rover from earth using a video camera. It was built by Mechanical Engineering graduate student James L. Adams. At that time vehicle had no on-board computer and was connected to a control console through a long cable and carried a TV camera. By 1971 Rodney Schmidt a PhD student working with Les Earnest, was able to make this cart follow a high contrast white line under controlled lighting conditions using video images processed by KA10 processor. Finally Hans Moravec was able to use binocular vision (single camera was

[1] In this thesis we use the terms *autonomous vehicle, intelligent vehicle and driver less cars* in the same sense.

[2] `http://www.stanford.edu/~learnest/cart.htm`

slid on a slider platform to get images from different positions) to navigate slowly around obstacles in a controlled environment. In 1979 the cart successfully crossed a structured chair-filled room without human intervention in about five hours (Moravec, 1980).

In 1977 Japan's Tsukuba Mechanical Engineering Lab vehicle followed a road up-to 50 meters at a speed of *30 km/h* by tracking white street markers using a video camera[3]. Although details are not available about their work however it looks that they used background subtraction techniques to follow white color.

Shakey (1966 to 1972), developed by Artificial Intelligence Center at SRI International (then Stanford Research Institute) looks to be the first robot using multiple sensors. It used three sensors, a TV camera, an optical range finder and tactile sensors. TV camera and Range finder sensors were used primarily for perception and path planning whereas tactile sensors were used for bump detection. *Shakey* worked in an indoor structured environment (Nilsson, 1984).

The vehicles mentioned so far had very limited perception capabilities and mainly used simple techniques for vision sensors for functioning. Moreover they worked in highly structured indoor environment having objects, walls and floors of very specific colors that could be easily recognized in images. They moved very slow due to the limited processing resources available. The main objective of these systems looks to prove the idea that artificial sensors and processing resources can be used to observe and understand (called perception) the world, an imperative prerequisite for autonomous navigation. Advancements in both hardware and software technologies resulted in more intelligent systems as given below.

In 1980s Ernst Dieter Dickmanns, a former German professor, was recognized as a pioneer of intelligent cars because of the tremendous amount of work he did with his team. Using vision technique along with probabilistic approaches like Kalman filters, he was able to build cars that could run on empty roads autonomously. By 1987 his intelligent vehicle *VaMoRs* could run at speeds up to *96 km/h*. This stimulated European Union to spend almost $1 billion for the pan-European Prometheus project (1987-1995) to conduct research in building autonomous cars. The result of this project had two culmination events: The first, in 1994, demonstrated two robot cars *VaMP* and *VITA-2* drove more than one thousand kilometers on a Paris multi-lane highway in standard heavy traffic at speeds up-to *130 km/h*, the second in 1995 a S-Class Mercedes-Benz car did a round trip from Munich to Copenhagen, going up-to speed of *175 km/h* on German Autobahn. These cars used vision sensors with computer vision techniques

[3] `http://www.tech-faq.com/driverless-car.html`

to achieve these goals. Although results look quite promising yet there were many limitations. For examples these vehicles could only run on a road in a lane and could detect vehicles ahead of them, however lane changing and handling traffic situation in intersection like scenarios was not possible. Moreover human input was often needed even in a lane on highways.

Later on a series of DARPA grand challenges attracted many people from around the world to put research efforts in autonomous vehicle driving, the winning teams of its 2007 challenge were able to complete 96 km long urban track in less than 5 hours. These vehicles are very close to the ideal intelligent vehicles however they were able to achieve these goals by using many redundant sensors such that the total cost of sensors was many times the cost of vehicle.

Recently VisLab's[4] VIAC challenge (4 autonomous vehicles followed a man driven vehicle from Italy to China with main purpose of getting datasets of different environments) and Google Cars have set new milestones in autonomous vehicle driving. Although VisLab vehicles used only vision cameras as main perception sensors however Google cars are using multiple sophisticated sensors (velodyne, radar, vision) for environment perception.

As of today all the intelligent vehicle systems mentioned are not capable of dealing with all possible scenarios that we may encounter while driving, especially on busy city roads. The reason being the amount of information required, about the environment, to deal with such situations successfully is yet a challenge to get from the sensors. Now it is a general acceptance that one single sensor, due to its limitations and uncertainty (given in section 1.3), is not capable enough to provide all necessary information required for this complicated task. Using multiple sensors is a potential solution, however combining information optimally from these sensors is a challenge.

Summarizing we can say that the degree of intelligent vehicle's autonomy depends on its capability of understanding external environment using its sensors' data, we term this capability as vehicle perception. More specifically, perception is the key step in the overall structure of an intelligent vehicle and all the other high level decision making is based on it. In the following we describe more precisely what it is all about and what kind of problems related to perception must be addressed.

[4] www.vislab.it

1.2 Intelligent vehicle perception

Literally perception (from the Latin perceptio, percipio) is the process of attaining awareness or understanding of the environment by organizing and interpreting sensory information. In the context of robotics (or intelligent vehicles) it involves getting data about the surrounding environment using sensors, processing these data and inferring an internal representation (we will call this internal representation as internal model) of this external environment.

In our context perception involves getting following important information from the sensor data:

- Position of objects w.r.t each other and shape of the road. These information are called map of the environment.
- Position of the intelligent vehicle in this map.
- Distinction between static and dynamic objects with trajectories of moving objects.
- Types of objects (cars, bicycles, persons etc) called classification information.

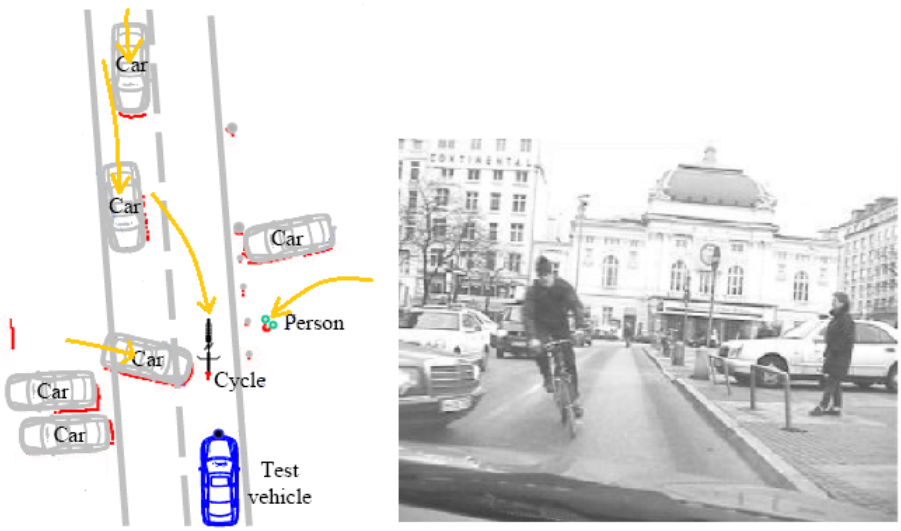

Figure 1.1: A goal of the robotic perception. Right: a real road scenario. Left: vehicle's internal belief about it.

Figure 1.1, from (Fuerstenberg *et al.*, 2002), is an attempt to show this goal of the intelligent vehicle perception. Image in the right side shows a road scene whereas left side shows the internal model of the intelligent vehicle inferred from sensor data. Data from laser sensor is shown as red dots whereas intelligent vehicle (also called host vehicle,

ego vehicle, test vehicle or demonstrator vehicle) is shown in blue color. It looks from this figure that host vehicle has inferred shape of the road, position and type of other objects in the environment, distinction between static and dynamic objects with trajectories of dynamic objects (shown as yellow arrows) and its own position w.r.t other objects in the environment, from sensor data (laser data in this case). Host vehicle usually needs to process multiple data frames to infer these information.

Perception is one of the most important tasks a robot must accomplish in order to function without human intervention. Perception, as defined above, requires to solve following sub problems:

- *Mapping*, defined as the process of establishing spatial relationships among stationary objects. It is necessary to build on-line maps using sensor data because due to the presence of temporary stationary objects a priori maps may not fulfill the purpose. Moreover it may be costly to get a priori maps for the target environment.
- *Localization*, defined as the process of establishing spatial relationship between vehicle and stationary objects in the environment. Intelligent vehicle needs to know its own position in the map.
- *Detection of moving objects*, it requires the vehicle to distinguish between static and dynamic objects.
- *Tracking of moving objects*, defined as the process of establishing the spatial and temporal relationships between moving objects, host vehicle and stationary objects.
- *Classification of objects*, it involves knowing the type of each object especially the dynamic objects.

First four sub problems are grouped in two problems: *SLAM* (Simultaneous Localization And Mapping) and *DATMO* (Detection And Tracking of Moving Objects) (Figure 1.2). While *SLAM* gives the robot the map of the environment where it is operating with the position of the robot in this map, *DATMO* allows it to be aware of moving objects in this environment so that appropriate actions may be taken while completing a task. Classification information can be considered as details inferred from sensor data to improve the results of these two problems. We will, especially, need them for *DATMO* part. Some preprocessing of sensor data may be needed before *SLAM*, like coordinates transformations for data representations etc.

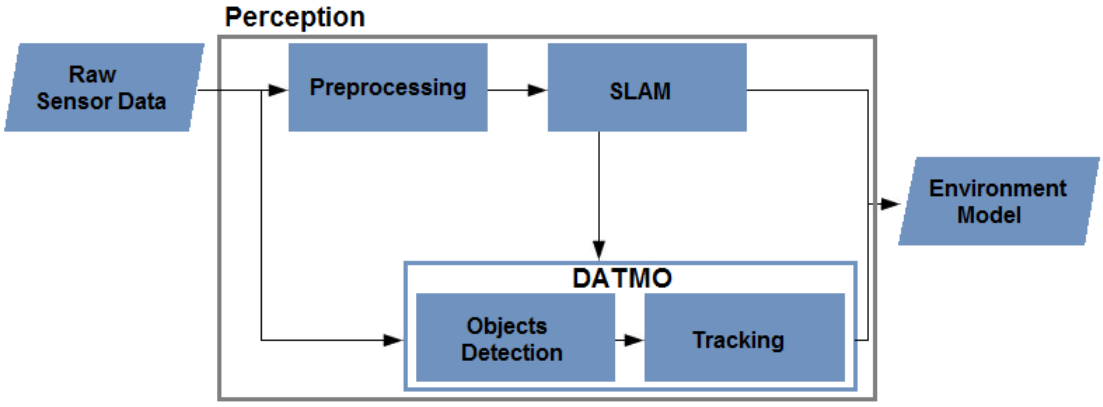

Figure 1.2: Inputs to perception consist of sensor measurements and control inputs (usually taken as the motion measurements like odometry or IMU). Common elements of perception output include: robot or vehicle state (its position in the map, its velocity, acceleration etc), map of the environment (details of the map depend on the target application) and list of moving objects with attributes like position, velocity etc. These elements are collectively called environment model.

Map details along with dynamic behavior (velocity, acceleration etc) of moving objects given by *SLAM* and *DATMO* solutions, are used by high level applications to assist the driver or to control the vehicle movement.

If we are able to accomplish both *SLAM* and *DATMO* reliably in real time, we can use the generated environment model to detect critical situations to warn the driver in advance or apply brakes automatically in case of inevitable collision to mitigate damage.

Finally the importance of perception can also be seen from the fact that it is the first element in the usual three component based robotic system architecture. They include: *Perception*, *Decision* and *Action*. This is usually called *PDA* robot architecture (shown in figure 1.3), the acronym of its three components. According to this architecture robot's actions are based on its decisions which in turn are based on the environment model given by perception.

Many researchers have worked on perception problem using different sensors in different scenarios. In this dissertation we have used two main sensors: Laser and vision (mono and stereo vision) to solve this problem. Especially we have developed techniques to fuse data from these sensors at different levels to improve moving object tracking sub problem for autonomous vehicles in crowded intersection like scenarios.

Since a solution to perception problem very much depends on the types of sensors used as input sources, it is important to know their characteristics. In next section after

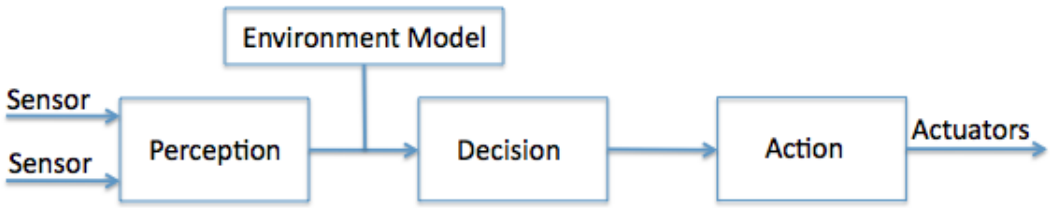

Figure 1.3: Robot decisions are based on the environment model given by perception module, hence perception is the base of robotic decisions and actions in PDA architecture.

giving a general introduction of sensors we detail the two sensors that we have used in our work.

1.3 Sensors

A sensor is a device that measures a physical quantity and converts it into a signal which can be read by an observer, an instrument or a computer. Sensors are the only way to get direct information from the environment. Sensors use physical properties of matter and laws of physics to perform their function. They are capable of perceiving a physical property or environmental attribute, such as heat, light, sound, pressure, magnetism or motion. To be useful, the sensor must map the value of the property or attribute to a quantitative measurement in a consistent and predictable manner. Usually a sensor measures a single aspect of the environment, so to get information about different aspects of the environment different sensors are used. Modern sensors even have data processing capabilities to transform observed data into more usable forms before outputting for the target system. The choice of sensors very much depends on the context of the problem to be solved. In the field of robotics in general and intelligent vehicle driving in particular the sensors which can be used to create an accurate internal representation of external environment are chosen. Examples include: 2D and 3D Laser scanners, Radars, Vision sensors (mono and stereo vision), IMUs, GPS etc.

These sensors can broadly be divided into two categories: passive and active. A *passive sensor* only measures the energy emitted by the objects in the environment to observe them whereas an *active sensor* emits energy and then observes the change in the reflected energy to estimate the state of the environment. Examples of passive sensors include photographic camera, infrared camera and radiometers etc whereas laser scanner, radar and ultra sonic are a few examples of active sensors.

Two concepts related to sensors: *sensor uncertainty* and *sensor model* are very important

to consider. Without information about these concepts for each sensor it is difficult to model correctly their behavior in the target system. Below we define what these concepts are. We end this section with a short description of the laser scanner and stereo vision camera mounted on the demonstrator used to get data for this work (left camera from this pair of stereo serves as mono vision sensor for us).

1.3.1 Sensor uncertainty

Sensor measurements are inherently uncertain, which means that they can only give an estimate of the measured physical property. Due to physical constraints it is impossible to make a perfectly error free sensor. Sensor uncertainty is caused by many types of errors (Mitchell, 2007) of which following two are commonly seen in many sensors:

- Random errors, these are caused by measurement noise and occur in an unpredictable and non repeating manner. They may be caused due to fluctuations in the capacity of the resistance of the electrical circuits in the sensor, or due to the limited resolution of the sensor.
- Systematic errors, they repeat themselves in a systematic manner and include following types of errors:
  - Calibration errors, they are caused by calibration process (the process of estimating sensor parameters) and usually are due to linearization of the calibration process for devices exhibiting non-linear characteristics.
  - Representation errors. Caused by the transformations from sensor space to a common representation format.

1.3.2 Sensor model

In order to analyze and use the output of a sensor it is important to model its behavior, this modeling of behavior is called sensor model and is defined as the description of the probabilistic relation between sensor measurement and actual state of the space (Yguel *et al.*, 2006). Formally if Z represents the sensor output and X the world state then the probability $P(Z|X)$ represents the sensor model and gives the probability of sensor output for a given world configuration. An important related concept is inverse sensor model represented as $P(X|Z)$ and gives the probability of a world configuration given a measurement. Usually a problem represented using one of these forms can be converted to the other by applying Bayes theorem.

1.3.3 Sensors used: LIDAR and Vision Camera

As mentioned earlier, our work in this dissertation focuses on LIDAR (LIght Detection And Ranging, but throughout the text we will write it as lidar or Lidar or simply laser scanner) and vision camera (mono and stereo) as main sensors to solve perception problem in the context of intelligent vehicles. The sensors mounted on the Volkswagen (VW) demonstrator used to get data sets for this work are shown in figure 1.4 along with their relative fields of view (FOV) and effective range information. Although there are other sensors (like short range radars and long range radar) also installed on the demonstrator however our work is concerned with lidar and stereo vision because field of views of 4 SRR (Short Range Radar) have no overlap with each other and very little overlap with lidar. Similarly field of view of LRR (Long Range Radar) is very narrow and is more relevant for a cruise control application. Although lidar covers the significant area in-front of the vehicle and is the primary perception sensor, information from stereo vision are very useful especially when vehicle is moving in crowded urban environment or when lidar is more noisy. In these situations an object occluded (or partially occluded) for lidar may be visible to stereo due to its different installation position on the host vehicle, or lidar may give many false positives due to noise that may not be the case with stereo vision due to its different functioning principle. Moreover information from lidar and Stereo vision are complementary with lidar providing precise position information of objects and stereo vision giving classification information with position estimates. Although for most of our work in this thesis we have used data sets from this Volkswagen demonstrator, however, for our work of low level fusion between laser and mono vision we have used data sets from a simulated demonstrator vehicle called *Sycab* developed by INRIA, because we needed colored images with high resolution laser scanner. More details about the simulated demonstrator and sensors installed on it will be given in 4th chapter. Below we discuss some important aspects of our main sensors installed on Volkswagen demonstrator vehicle.

Lidar

Lidar sensor used on this demonstrator is a *Hella IDIS 2.0* that is capable of allowing different configurations of use. With highest resolution configuration it has a horizontal field of view of 160° and maximum resolution of 1°. It has 161 laser beams with each beam capable of detecting two targets with vertical opening angle of 3°. Two targets means that the beam first scans in horizontal plane then in a plane at an angle of 3° from horizontal plane. If the detected object in second case is different from the first

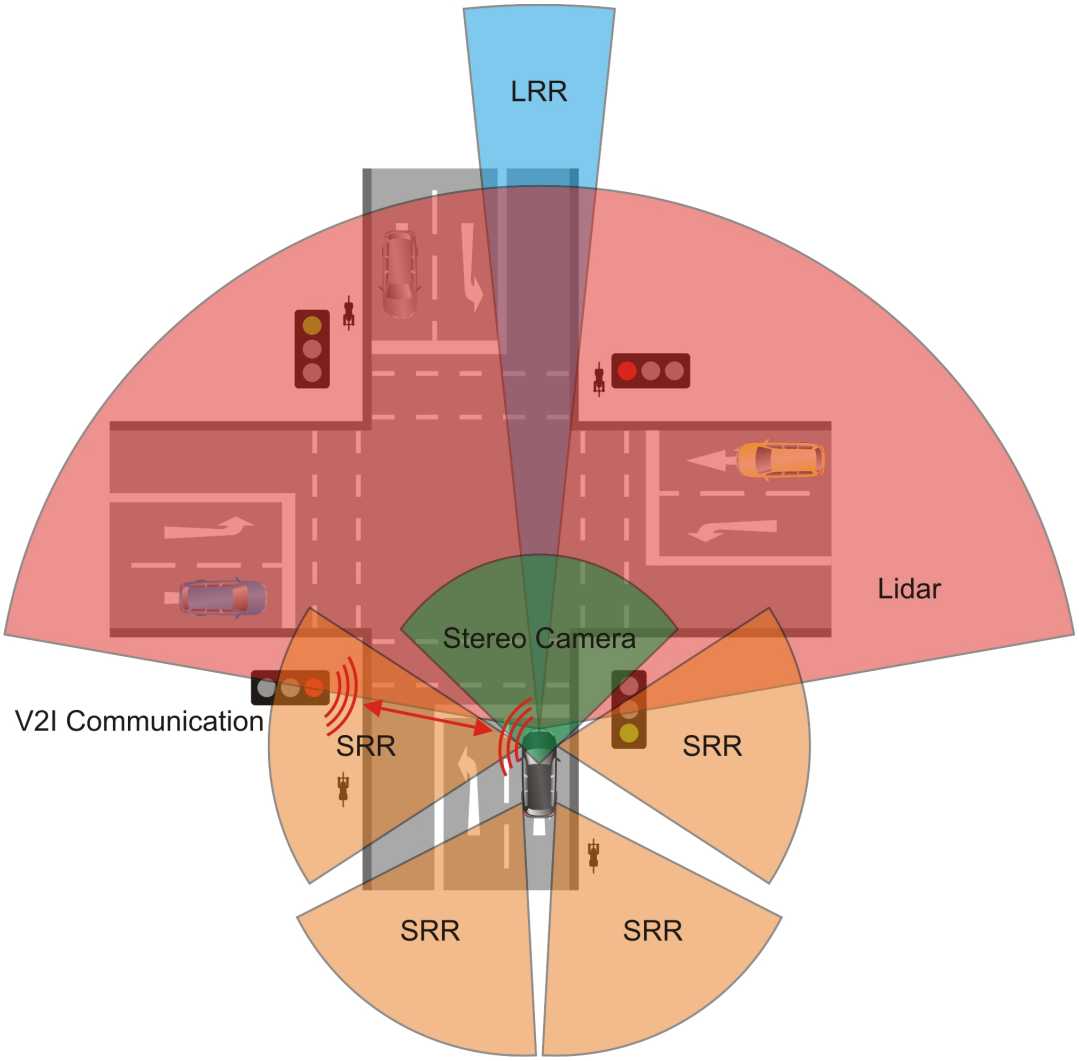

Figure 1.4: Sensors mounted on the demonstrator vehicle. Our work in this thesis is only concerned with *Stereo Camera* (green FOV) and *Lidar* (pink FOV). Other sensors include four short range radars (SRR) one long range radar (LRR) and vehicle to infrastructure communication setup (V2I).

one (using some threshold on difference of distance between them) it is reported as a second target. In our work we only consider the closest target. However, to reduce the amount of data to be transferred on the communication layer between different modules and to ensure a minimum 20 fps (frames per second) rate for all the sensors Volkswagen decided to use a low resolution configuration for this lidar sensor. In this configuration lidar has 2° resolution with 160° field of view. So it has 81 laser beams with each beam capable of detecting two targets as explained above. All our data sets have this later configuration of the lidar sensor. In both configurations the range is from 1 meter to 150 meters, however, with 2° resolution hits detected beyond 50 meters are so far apart that only one hit is observed on an object making it difficult to decide

whether its a true object or false positive. To avoid this problem we reduce the effective range to 50 meters. Practically observed lidar scan rate is 16.6 Hz.

Sensor model and inverse sensor model for a laser scanner is shown in figure 1.5. It is important to mention that at the time of registering data sets for current work this laser scanner had low signal to noise ratio. Other than usual uncertainties it showed shaky beam behavior and erroneous reflections from white lane markings on the road. Because of this perception solution using only laser showed many false positives.

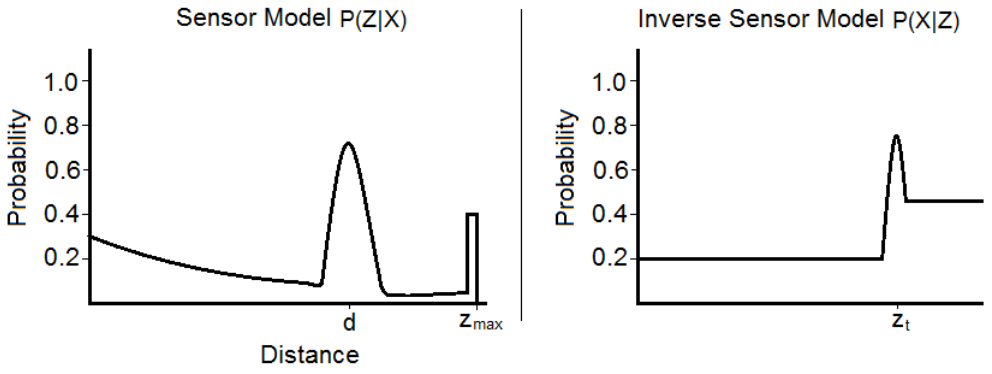

Figure 1.5: Left: sensor model of a laser scanner, the curve models the probability of getting a hit at different distances for an object at a distance of d (ground truth), z_{max} is the maximum range of the scanner. The non null probability on other than d location is due to sensor uncertainty mentioned above. Right: inverse sensor model of the same laser scanner, this curve models the probability of presence of an object at different distances for which laser gives distance z_t.

Lidar is one of the main sensors being used today for intelligent vehicle perception (and for robotics in general) because of its high accuracy of obstacle distance measurement. High resolution laser scanners have been successfully used for intelligent vehicle perception, for example by Vu (2009) and Wang (2004). Moreover it performs equally well both during day and night and is not affected by different weather conditions. This makes lidar a preferred choice for intelligent vehicle perception. However one shortcoming of this sensor is its inability to give object classification information, moreover if resolution is not high ($< 1^\circ$) or sensor uncertainty is high then there are many false positives making overall perception results very poor. These shortcomings can be compensated by adding another complementary sensor.

Stereo Camera

This stereo vision system has two *JAICV-M4+* cameras with each camera capable of taking a gray scale image of high resolution at 24 fps (frames per second) rate. The baseline length is 340 mm. Images from left camera are used for our mono vision work. An example stereo vision camera and how it works is shown in figure 1.6.

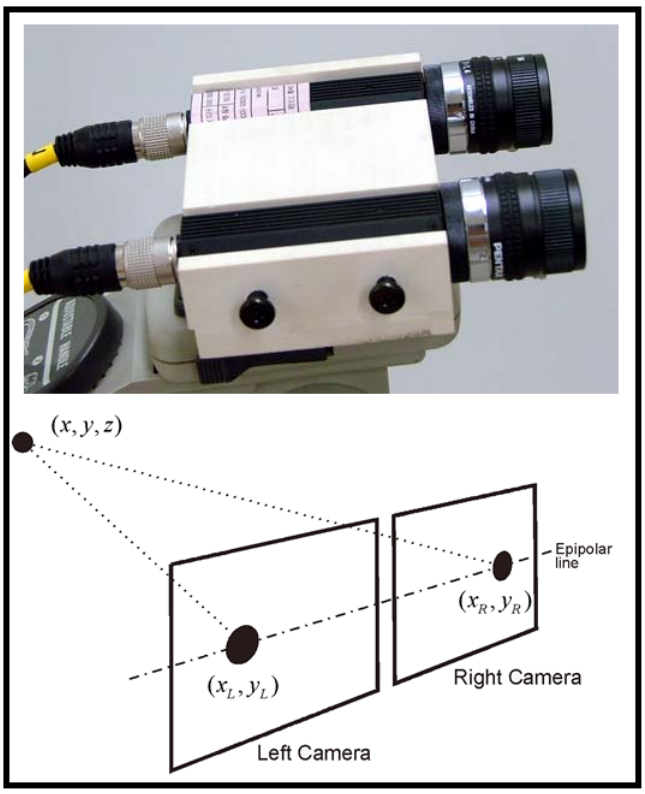

Figure 1.6: A stereo vision system (top) and how it works (bottom). A point in world (x, y, z) is projected on the image planes of both cameras, using epipolar geometry an estimate of its distance can be made.

Stereo vision processing was done by a team of Technical University of Cluj-Napoca Romania[5] (Nedevschi *et al.*, 2009a,b). We had access to two levels of data from the output of their processing: i. *3D points* with each point having position information from a fixed frame of reference and ii. *classified objects list* with each object having information about its position, dimensions (corner points of its bounding cuboid) and class (pedestrian, vehicle, pole, curb, unknown). No motion information of objects were provided. Since we get preprocessed data from stereo vision we do not know the direct sensor model and inverse sensor model. However in-order to perform fusion with laser we have estimated measurement uncertainty for stereo vision indirectly, more details

[5] `http://users.utcluj.ro/~nedevski/`

about this will be given in third chapter. Practically observed processed stereo data frame rate is about 12 Hz.

Vision sensors are the very first sensors used for intelligent vehicle perception. In fact the first autonomous vehicles developed by Ernst Dieter Dickmanns and his team only used vision sensors. Both mono and stereo vision sensors have been used by many researchers for both indoor and outdoor environment perception (Danescu *et al.*, 2009, Mittal and Davis, 2003). Although its performance heavily depends on lighting and weather conditions, it still is a popular sensor being used for vehicle perception because of its low cost and rich information.

From the characteristics mentioned both for laser and vision sensors our hypothesis in this thesis is that by using lidar and vision camera (both mono and stereo) as perception sensors we can devise fusion techniques at different levels of abstractions to improve perception results. In next section we explain what multi sensor data fusion is along with brief description of abstraction levels where we have performed fusion for these sensors.

1.3.4 Data synchronization

In Volkswagen demonstrator each data frame from each sensor or module has a time stamp that is used for synchronization. We save some frames of these sensors in arrays and a simple search based on this time stamp gives the closest matching data frames from these sensors for fusion. We have seen that size of these arrays never exceeds from 4.

1.4 Multi-sensor data fusion

Multi-sensor data fusion is the process of combining sensory data or data derived from sensory data in such a way that resulting information (fused data) is in some sense better than data from individual sensors. The fusion may take place between similar (homogeneous) or different (heterogeneous) sensors. Sensors measuring same properties of the environment are called homogeneous sensors whereas the ones measuring different properties of the environment are called heterogeneous sensors. For example two laser scanners are homogeneous whereas a laser scanner and a camera are heterogeneous sensors. The rationale to use homogeneous sensors for observations is either to reduce inherent sensor uncertainty or to increase sensor field of view. In either case an effective data fusion scheme is necessary to make optimal use of the sensors.

In heterogeneous sensors case, complementary information are obtained from different sensors, so an effective fusion technique will result in more complete information about the environment under observation. Although our main sensors are heterogeneous but they have some homogeneous aspects as well (for example stereo vision also gives depth information of detected objects like laser scanner). Figure 1.7 elaborates this fusion concept.

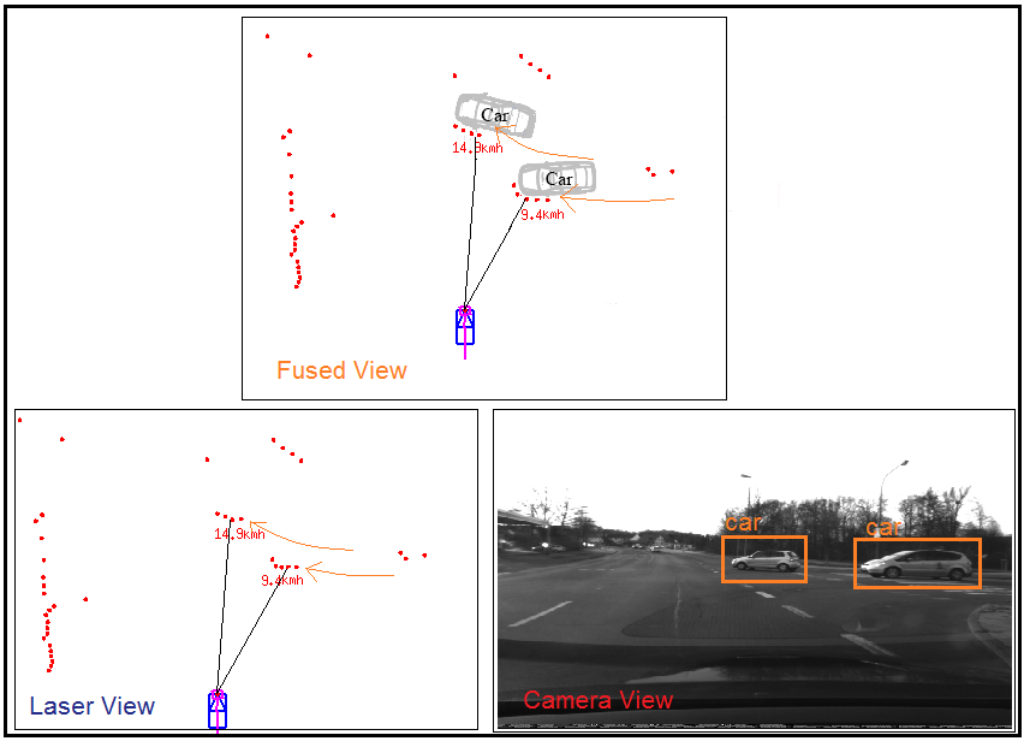

Figure 1.7: Fusion example between laser and mono vision. Laser (bottom left) detects moving objects, calculates their position and velocity information whereas vision (bottom right) classifies them as cars. After fusion (top), system knows that it is tracking two cars.

Multi sensor data fusion has been a topic of research since long especially in the field of defense. Main fusion applications developed by people in this field are related to ocean and air surveillance and battlefield intelligence. Non military applications are related to intelligent robots, disease diagnostics, environment monitoring, state monitoring of industrial setups/equipments etc. Hall and Llinas (2001) have detailed many application fields of multi sensor data fusion. Below we describe the abstraction levels where we have performed fusion to improve intelligent vehicle perception

1.4.1 Fusion for intelligent vehicle perception

Extracting information from sensor data can be termed as finding features of interest from these data. For example from laser data we can extract features like position and shape (lines). Similarly from vision sensors we can get more abstract features like class of the objects. Intelligently combining these features from these sensors may give more complete view of the objects and the environment and is the objective of a fusion system.

Our work in this dissertation is related to the development of fusion techniques for data from vision and laser sensors at different levels for intelligent vehicle perception to improve perception results, more specifically to improve tracking of moving objects. We have identified four levels in *SLAM* and *DATMO* based perception architecture (shown in figure 1.2) where fusion can take place. Same perception architecture with position of identified fusion levels is shown in figure 1.8. These identified fusion levels are:

- *Low level* with minimum preprocessing (like projecting data into a common frame of reference or classifying individual pixels etc) data from each sensor is represented in compatible forms and these individual representation forms are combined to get fused representation. Output of fusion at this level is input to the *SLAM* module. We have performed fusion at this level between laser and mono vision and laser and stereo vision. Data set for this level of fusion between laser and mono vision is obtained from Cycab simulated demonstrator as mentioned earlier, however for all other purposes data from real Volkswagen demonstrator vehicle have been used.

- *Map level*, output of each sensor is processed to solve *SLAM* problem and the maps generated by individual sensors are combined to get a fused map. This level is more interesting for robots moving in cluttered environments, however for intelligent vehicles moving on roads map generated by laser scanner is sufficient so we have not considered this level in our work as explained in next chapter.

- *Object detection level*, output of each sensor is processed to detect objects in the environment, then we perform fusion between these lists of objects to get a list of fused objects, fused objects have more complete properties than their pre fusion versions. Fusion at this level is done between laser and mono vision and laser and stereo vision.

- *Track level*, fusion at this level involves combining lists of moving objects detected and tracked over time by individual sensors. Objective of this fusion is to reduce the number of false tracks. We have performed this fusion between laser and stereo vision.

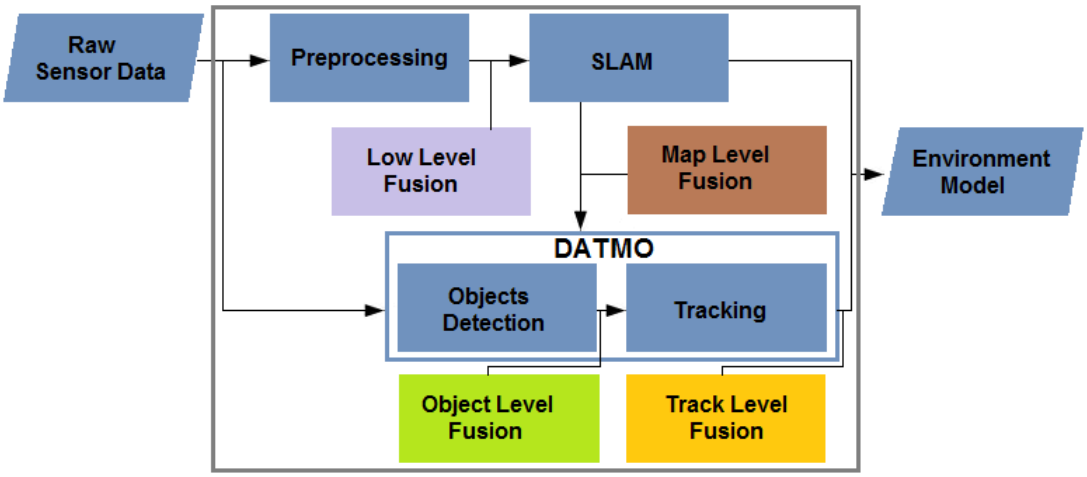

Figure 1.8: *SLAM* and *DATMO* based perception architecture with identified fusion levels.

Detailed architecture of each of these levels is given in next chapter.

1.5 Contributions

In this dissertation we have implemented and validated the *SLAM* and *DATMO* solutions developed by Vu (2009) using a new laser scanner having different specifications than the one used in original work. Furthermore we have used additional sensors, stereo vision camera and mono camera, and have devised fusion techniques at different levels to improve the results of above solutions. The main contributions of this research are as follows:

First contribution of this research is the development, implementation and comparison of fusion techniques at three levels of abstraction: *Low level*, with minimum preprocessing of each sensor we represent their output in a common format (occupancy grids to be precise) then using fusion technique we combine them in a single representation which is further used as input to the next perception step. *Object detection level* and *Track level*, as given above, we have developed techniques to solve the object/track associations and then to fuse the properties of these associated objects and tracks. The result of these fusion techniques are improved perception, especially *DATMO* , results.

Second contribution is the development of a fast technique for detection and tracking of road borders in noisy laser data. The output of this module, a road model, is used in *DATMO* module to reduce the number of false moving objects[6] detected due to noisy laser data.

Third contribution is the integration of the fusion techniques on a real Volkswagen demonstrator vehicle. This integration has been performed in the framework of a European Project INTERSAFE-2[7]. This project is concerned with the road user safety on intersections. The objective of this project was to develop a complete safety solution for all types of road users on intersections using on-board and infrastructure sensors. Since multiple sensors were used on the demonstrator vehicle so we developed fusion techniques to accomplish the perception part of this project on demonstrator vehicle.

1.6 Thesis layout

Rest of this dissertation is structured as follows: In next chapter we give a detailed state of the art of robotics perception especially *SLAM*, *DATMO* and multi-sensor data fusion. In this chapter we also explain the schema of three levels for which we have developed fusion techniques in the context of autonomous vehicle driving. Chapter 3 details the fusion process between laser and stereo vision camera along with our road border detection technique that we have developed to reduce false alarms in noisy laser data to improve *DATMO* results. In chapter 4 we give fusion details between laser and mono camera. Chapter 5 details the INTERSAFE-2 project with contribution of the partners involved. In last chapter we conclude our work with some perspectives.

[6] A false moving object is one which actually does not exist but due to sensor uncertainties is inferred from its data.

[7] `www.intersafe-2.eu`

CHAPTER 2

Vehicle Perception with Fusion - State of the Art

2.1 Introduction

As discussed in the previous chapter, perception process in intelligent vehicles is concerned with building an internal representation of external environment by interpreting data obtained from its sensors. During this process the vehicle needs to estimate the relative positions (may include other details like shapes and classification information) of static objects, its own position in the environment and needs to find the moving objects and track them over time. We grouped these sub problems into *SLAM* and *DATMO*. This perception process takes some inputs and after processing gives some outputs. In order to discuss state of the art techniques developed to solve perception problem we need to formally define these inputs and outputs (shown in figure 2.1). Here Z_t, sensor inputs accumulated until time step t, and U_t, control inputs but

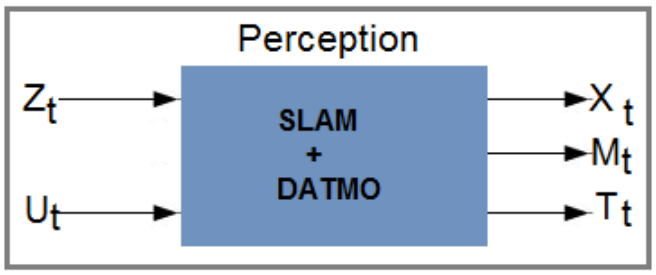

Figure 2.1: Perception inputs and outputs (see text for details).

usually taken equivalent to the motion measurements given by odometer or inertial measurement units (IMU), are given as:

$$Z_t \triangleq z_{0:t} = \{z_0, z_1, \ldots, z_t\} \tag{2.1.1}$$

$$U_t \triangleq u_{1:t} = \{u_1, u_2 \ldots, u_t\} \tag{2.1.2}$$

Where z_i is sensor input and u_i is motion measurement at time instant i.

The output X_t is the set of estimated vehicle states (position and orientation define vehicle state) until time t. So if x_i is the state of vehicle at time instant i then we have:

$$X_t \triangleq x_{0:t} = \{x_0, x_1, \ldots, x_t\} \tag{2.1.3}$$

The absence of u_0 means that first motion measurement becomes available only when vehicle has moved from state x_0 to x_1.

The map M_t is the set of static objects detected until time t and is represented as:

$$M_t = \{m_0, m_1, \ldots, m_k\} \tag{2.1.4}$$

Finally the set of tracks at time t is the set of moving objects o_i, detected consistently in some previous data frames, given as:

$$T_t \triangleq \{o_0, o_1, \ldots, o_m\} \tag{2.1.5}$$

Where m is the number of tracked dynamic objects at time instant t.

It is important to note, from these definitions, that a time t subscript with a capital letter variable means history (0 to t) of that variable whereas same subscript with a small letter variable means only one value for that variable.

A probabilistic solution to perception problem needs to estimate following a posteriori probability density:

$$P(x_t, M_t, T_t | Z_t, U_t, x_0) \tag{2.1.6}$$

It is a complex problem and is generally divided into following sub problems (*SLAM* and *DATMO* respectively):

$$P(x_t, M_t | Z_t, U_t, x_0) \tag{2.1.7}$$

$$P(T_t | Z_t, x_t, M_t) \tag{2.1.8}$$

Using multiple sensors to address perception problem, essentially, keeps the theoretical structure of the problem same. It adds new inputs and the result is supposed to be improved outputs or parts of outputs. Perception structure using inputs from two sensors: laser Z_t^l and vision Z_t^v is shown in figure 2.2. And the perception formalization equivalent to 2.1.6,in this case, becomes:

$$P(x_t, M_t, T_t | Z_t^l, Z_t^v, U_t, x_0) \tag{2.1.9}$$

In the following we present state of the art perception solutions, first, without fusion and then with fusion. We detail the different fusion levels that we have tested in this

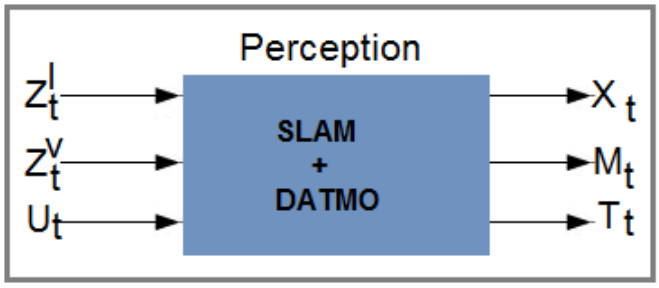

Figure 2.2: Structure of perception problem using inputs from two sensors: laser and vision.

dissertation in section 2.3.2 and cite the relevance of our work and the improvements caused by it. Section 2.4 gives the architecture of our road border detection module and its position in the *SLAM* and *DATMO* based perception architecture, we also discuss some related works in this section. We give a short synthesis of the chapter in the last section.

2.2 Perception without fusion

Since we have divided perception into *SLAM* and *DATMO* parts we will discuss literature review on them in this order.

2.2.1 SLAM

Simultaneous Localization and Mapping (*SLAM*) also called Concurrent Mapping and Localization (*CML*), a fundamental problem of autonomous navigation, has been an important field of research for robotics community. In simple terms it means, is it possible for a robot (placed at unknown position in an unknown environment) to incrementally build a consistent map of the environment while simultaneously determining its location within this map using inputs from sensors? *SLAM* is a chicken-egg problem because to find the position of the vehicle we need to know the exact map and to build the map of the environment we need to know exact position of the vehicle in the environment. The importance of this problem can be imagined from the fact that a solution to *SLAM* problem was considered as a 'holy grail' by robotics community because it would provide the means to make a robot truly autonomous (Durrant-Whyte and Bailey, 2006). In the following we first give a short description of the progressive development of techniques addressing *SLAM* problem, then we summarize formalizations of modern solutions to this problem.

In retrospect - development of *SLAM*

Before the localization and mapping were taken as problems to be solved simultaneously the early navigational systems showing some autonomy can be divided into following types:

- *Line following*; these systems used sensors to observe very specific "features" in the environment and followed them to perform their assigned tasks. For examples vehicles in industrial setup following current carrying wires buried in ground using induction coils (Tsumura, 1986) and odor (Deveza *et al.*, 1994, Russel *et al.*, 1994) or heat (Russel, 1993) trail following systems. No on-line mapping and localization was addressed in these systems. They were simple navigational systems.

- *Beacon based systems*; They were primarily localization systems based on observing artificial beacons installed in the environment, using sensors. From the observed position information of the beacons they were able to calculate their own position in the environment (Kleeman, 1992) and (Durrant-Whyte, 1996). Beacons can be active or passive, active beacons emit signals which are received by the robot whereas passive beacons do not emit signals, robots use active sensors to observe them. GPS (Global Positioning System) based localization (Evers and Kasties, 1994) is also a form of active beacons. Other than GPS systems beacons are, in general, costly to install and maintain, whereas GPS based systems suffer from the problems of accuracy and unavailability of signals due to different reasons (Durrant-Whyte, 1996, Poppen, 1993).

- *Localization on a priori maps*; For example Zhang (1994) and Vandorpe *et al.* (1996) have described systems where robot extracts features from sensor data to localize on a priori input maps. In contrast to beacons, features are naturally occurring distinctive markers (landmarks) in the environment.

The main disadvantage of these systems is that they cannot be used in unknown environments because either they need specially modified environments (having artificial beacons installed) or a priori maps for their functioning.

The work on *SLAM* started with seminal works by Smith and Cheeseman (1987), Smith *et al.* (1990) and Durrant-Whyte (1988) in which they established a statistical basis for describing uncertain relationships between landmarks and manipulating geometric uncertainty and measurement uncertainty described by Brooks (1985). These works are a basis of modern *SLAM* solutions now. An important conclusion of Smith *et al.*

(1990) was that the estimates of landmarks observed by the robot are correlated because of the common error in estimated vehicle location. As a result of this conclusion positions of all landmarks were made part of the state vector requiring update on each observation. Inevitably the computational and storage complexity increases as robot moves and explores new areas by finding new landmarks (Hebert *et al.*, 1996). So to keep the size of state vector in limits some authors either used thresholds (Leonard and Durrant-Whyte, 1992) or did not take these correlations into account altogether (Rencken, 1993).

According to Durrant-Whyte and Bailey (2006) the work by Smith *et al.* (1990) did not look at the convergence properties of the map, so at that time it was assumed that estimated map errors display random walk behavior so people tried to solve localization and mapping problems separately. However it was realized that if mapping and localization are formulated as a single estimation problem then they actually converge. These convergence properties were first investigated by (Csorba, 1997) in his PhD thesis. The acronym 'SLAM' was first introduced in a paper presented at 1995 International Symposium on Robotics Research (Durrant-Whyte, 1996).

Above mentioned works primarily used landmarks extracted from sensor data to solve *SLAM* problem, however Elfes (1989a) introduced occupancy grids (OG) framework for localization and mapping. In this framework the space is divided into cells where each cell has a probabilistic estimate of its occupancy state. Usually a high value of cell probability indicates the cell is occupied and a low value means the cell is empty. Although Kriegman *et al.* (1990) argues that OG cannot cope with the uncertainties inherent in the data available to the robot yet this framework has been successfully used for localization and mapping (at least locally) (Elfes, 1989b, Konolige, 1997, Vu *et al.*, 2008, Vu, 2009).

Modern solutions

Modern solutions of *SLAM* problem are probabilistic in nature and estimate the joint probability given in equation 2.1.7. Using Bayes formula and Markovian assumptions this equation can be expanded as:

$$\underbrace{P(x_t, M_t | z_{0:t}, u_{1:t})}_{\text{posterior at t}} \propto \underbrace{P(z_t | x_t, M_t)}_{\text{update}} \underbrace{\int P(x_t | x_{t-1}, u_t) \underbrace{P(x_{t-1}, M_t | z_{0:t-1}, u_{1:t-1})}_{\text{prior}} dx_{t-1}}_{\text{prediction}} \quad (2.2.1)$$

This equation (2.2.1) gives the modern recursive two step, prediction (time-update) and correction (measurement-update) solution. More precisely these are given as:

Prediction (time-update)

$$P(x_t, M_t|z_{0:t-1}, u_{1:t}) = \int P(x_t|x_{t-1}, u_t)P(x_{t-1}, M_t|z_{0:t-1}, u_{0:t-1})dx_{t-1} \tag{2.2.2}$$

Correction (measurement-update)

$$P(x_t, M_t|z_{0:t}, u_{1:t}) = \frac{P(z_t|x_t, M_t)P(x_t, M_t|z_{0:t-1}, u_{0:t})}{P(z_t|z_{0:t-1}, u_{0:k})} \tag{2.2.3}$$

In these equations two factors: $P(z_t|x_t, M_t)$ and $P(x_t|x_{t-1}, u_t)$ are important and need some explanation.

The first factor $P(x_t|x_{t-1}, u_t)$ (vehicle motion model), used in prediction step, states how vehicle moves from previous state to current state under control inputs. In robotics, usually, instead of considering control commands, odometry measurements taken at current state are used to predict vehicle state.

The second factor $P(z_t|x_t, M_t)$, used in the update step, defines the probabilistic description of the sensor and is called sensor model. For an effective solution to *SLAM* problem it is important to define this sensor model for each sensor very carefully by taking into account all details and uncertainties of the sensor behavior in the target environment where this sensor will be used.

Map representation

The equations given above present a theoretical solution to the *SLAM* problem, however implementable solutions, from these equations, depend on how the map and probabilities are represented. Vu (2009) has divided the map representation techniques into following types (shown in figure 2.3):

- *Direct representation (point cloud)*; usually used for range sensors (like laser scanner), is a method of map representation using raw data without extracting features. Cole and Newman (2006) has used this representation for 3D *SLAM*. This is a simple method but lacks a precise representation of sensor uncertainty.
- *Feature based representation*; raw sensor data is processed to extract features which are used to represent map. This is a compact representation technique that depends on a set of distinguishable geometrical features extracted from sensor data. However, only a predefined set of features can be used, moreover, environments having irregular objects can be a challenge to represent. Dissanayake *et al.* (2001) have implemented a feature based *SLAM* solution.

- *Grid based representation*; the environment is divided into rectangular (or cubical for 3D) cells and the state of occupancy of these cells is inferred from the sensor data. Occupancy grids framework introduced by Elfes (1989b) falls in this category and has been used extensively to represent robot environment.

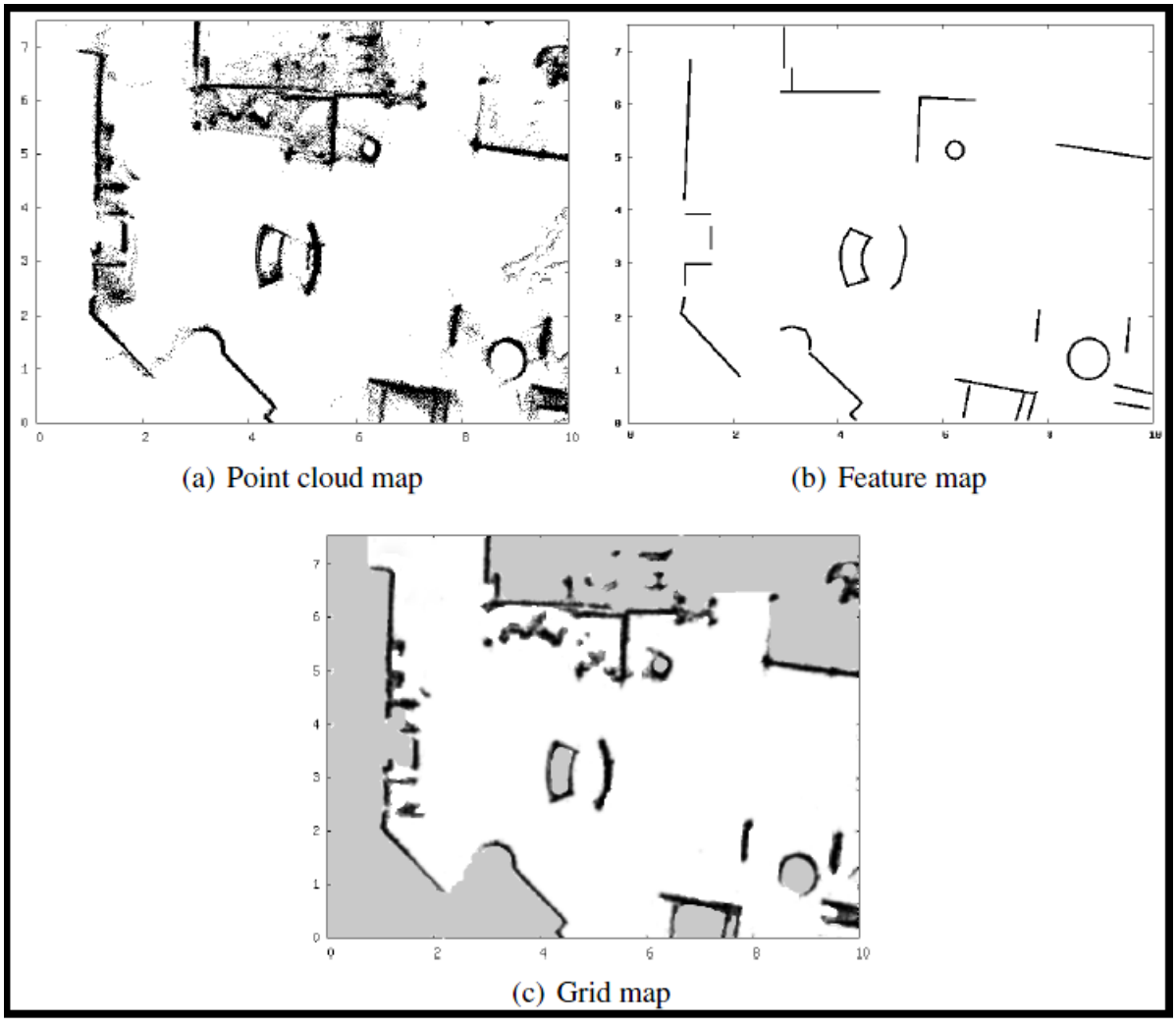

Figure 2.3: Map representation techniques used to represent same environment.

Usually *point cloud* representation is used for 3D maps, so in the following we will only discuss solutions for other two types.

Feature based solution

A preferred choice for *feature based* map representation is Kalman filter (Kalman, 1960) based *SLAM* solution if following assumptions are made:

- Vehicle motion model ($P(x_t|x_{t-1}, u_t)$) is linear with added Gaussian noise.
- Measurement model ($P(z_t|x_t, M_t)$) is linear with added Gaussian noise.

- Initial state uncertainty is Gaussian.

If these conditions hold then Kalman filter based solution is a standard recursive two step process consisting of prediction (time update) and correction (measurement update) steps given in equations 2.2.2 and 2.2.3 respectively.

Usually the linearity condition for vehicle motion model is not true and the new position x_t depends non linearly on previous pose x_{t-1} and control input u_t. In this case these non linear functions can be linearized using Taylor series expansion and the resulting filter is called extended Kalman filter (EKF) (Gordon *et al.*, 1993). For highly non linear and non Gaussian systems particle filter based solutions like FastSLAM (Montemerlo and Thrun, 2002) can be used.

Recently efforts have been made to merge other techniques with Kalman based solutions to address different problems for different scenarios or different sensors. For example Gérossier *et al.* (2009) have used Fourier-Mellin transform for microwave radar images to present an EKF based *SLAM* solution.

Grid based solution

For grid based map representation technique an incremental maximum likelihood based *SLAM* solution is more appropriate. In this formulation the maximum likelihood of the estimated vehicle state and environment map at time t is given as marginal likelihood:

$$< \hat{x}_t, \hat{M}_t >= \underset{x_t, M_t}{\arg\max} \{P(z_t|x_t, M_t)P(x_t, M_t|u_t, \hat{x}_{t-1}, \hat{M}_{t-1})\} \tag{2.2.4}$$

Here $\hat{x}_t$ and $\hat{M}_t$ are the latest maximum likelihood based estimates of vehicle state and environment map respectively. This equation (derived from 2.2.1) shows that the new maximum likelihood estimates of vehicle position and map depend on sensor model and previous estimates of position and map. However to solve equation 2.2.4 a good estimate of new vehicle position at time t can be obtained by searching in vehicle pose space using new measurement and previous map at time step $t-1$, mathematically it means:

$$\hat{x}_t = \underset{x_t}{\arg\max} \{P(z_t|x_t, \hat{M}_t)P(x_t|u_t, \hat{x}_{t-1})\} \tag{2.2.5}$$

A scan matching technique (Vu *et al.*, 2007) based on particle filter can be used to find best value of new vehicle pose that fits new measurement with map constructed so far. After we have found the best pose $\hat{x}_t$ then we can construct new map $\overline{M}$ using this pose value and the latest sensor measurement z_t. This new map can be combined with map estimate at time step $t-1$ to get new map estimate as follows:

$$\hat{M}_t = \hat{M}_{t-1} \bigcup \overline{M} \tag{2.2.6}$$

Equations 2.2.5 and 2.2.6 actually implement the *SLAM* solution given in equation 2.2.4. This completes the solution for *SLAM* problem.

The advantage of ML-SLAM is that it can be applied to any kind of representations whereas KF-SLAM works with feature based representation. Moreover ML-SLAM is fast because it calculates new map based only on current pose estimate and previous map (at time step $t-1$).

In this thesis we have implemented a maximum likelihood based solution (Baig *et al.*, 2009) using occupancy grid framework. Next we discuss state of the art solutions to the second part (*DATMO*) of the perception problem.

2.2.2 DATMO

Second important part of intelligent vehicle perception is the *Detection And Tracking of Moving Objects* (*DATMO*) present in the environment along with autonomous vehicle, and has been a hot field or research for decades (Bar-Shalom and Tse, 1974, 1975, Blackman and Popoli, 1999, Burlet *et al.*, 2007, Wang, 2004). A solution to *SLAM* problem either supposes that no moving objects are present in the environment or they are filtered out for further processing. Many of the modern tracking systems are based on the techniques developed for old air and ocean surveillance systems (Fortmann and Baron, 1979, Singer and Stein, 1971), these systems were primarily based on radar and sonar sensors respectively with the assumption that all the detected targets were moving (Fortmann and Baron, 1979, Singer *et al.*, 1974). However this is not the case when we are dealing with intelligent vehicles on the real road scenarios. Finding both static and moving objects is normal on roads, so the vehicle must be able to distinguish between them.

In *DATMO* , usually, detection of moving objects is handled separately from the tracking part and detected list of objects/targets is taken as input to the tracking part. In the following we follow the same order.

Moving objects detection

A huge amount of work has been done to detect moving objects, especially by vision community (Jain *et al.*, 1979, Li, 2000, Taleghani *et al.*, 2009, Zhang, 1994). These techniques have primarily been based on background subtraction or motion cues. Similar to background subtraction techniques have also been used in occupancy grids to detect moving objects (Burlet *et al.*, 2007, Vu *et al.*, 2009, Wang *et al.*, 2003). These techniques are based on inconsistencies observed for new data by comparing them with maps con-

structed by *SLAM*. Petrovskaya and Thrun (2009) and Vu and Aycard (2009) have also proposed model based techniques to detect moving objects on the roads in the context of autonomous vehicle driving. Recently Dempster-Shafer theory based grids (evidential grids) have been used to detect moving objects using conflict analysis techniques (Moras *et al.*, 2011a,b).

In our occupancy grid based solution if $M_t[z_t^i]$ is the cell of map M_t where ith part of measurement z at time t is projected (like ith laser beam of laser scan z_t) then the dynamic parts of the measurements is the set $\{z_t^i | M[z_t^i] \text{ is marked free}\}$.

Moving objects tracking

The detected moving objects are input to the moving object tracking (*MOT*) module. Most of the researchers have divided the *MOT* systems into four parts: 1) Gating (gate is an area around the next predicted position of a track where new observations of same track are searched) , 2) Data Association (is the process of assigning new observations to the existing tracks, this association takes place in gate area), 3) Track Management (is the process of keeping list of tracks and rules about creation of new tracks and deletion of old tracks) and 4) Filtering (a recursive process of estimating properties of tracks usually based on prediction and update steps) as shown in figure 2.4. These elements are even found in the early tracking systems for air traffic control or ocean surveillance. For example Singer and Stein (1971) calls the filtering as *track-update* and data association as *return-to-track correlation* and mentions the *Gates* based on predicted measurement for the object in track and statistical distance test. S.Blackman and Popoli (1999) have also given same architecture with an additional initiating element of *Sensor Data Processing and Measurement Formation* which in our case is equivalent to preprocessing done for detecting the moving objects. Two out of these four elements, namely *Data Association* and *Filtering* have attracted much more research efforts than *Gating* and *Track Management* this is because *Gating* is controlled by the predictions made by filtering and *Track Management* usually depends on the employed *Data Association* techniques. However a large number of researchers have developed many different techniques and their flavors for both *Data Association* and *Filtering* to solve *MOT* problem in either different scenarios or using different sensors. In the following we structure state of art according to techniques developed for these two elements.

Theoretically if ω is any set of tracks at time t then tracking is the optimal set of tracks $\omega^* = \underset{\omega}{\text{argmax}} P(\omega | z_{1:t})$

Data Association Techniques

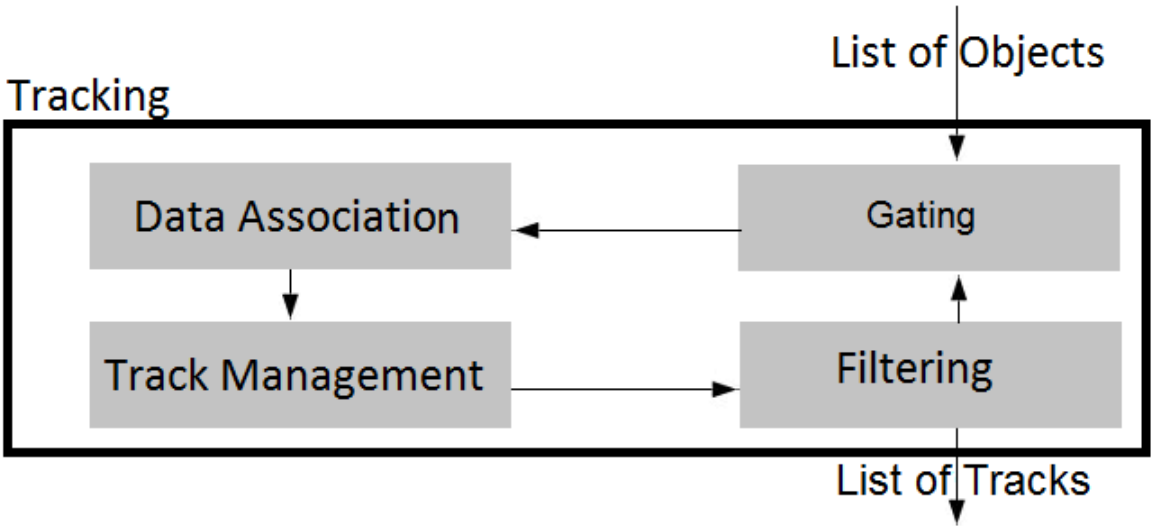

Figure 2.4: Typical elements of a Moving Objects Tracking system.

Some well known data association techniques are discussed below.

Global Nearest Neighbor (GNN) is the simplest data association approach that attempts to find and to propagate the single most likely hypothesis at each data scan. It has been used by many researchers as a data association choice (Crowley *et al.*, 1988, Deriche and Faugeras, 1991, Singer and Stein, 1971). The term global is used to refer to the fact that the assignment is made considering all possible (within gates) associations under the constraint that an observation can be associated with at most one track. This distinguishes GNN from the simple nearest neighbor (NN) approach in which a track is updated with the closest observation even if that observation may also be used by another track (Blackman, 2004). Konstantinova *et al.* (2003) has quantitatively evaluated the performance difference of these two flavors. This approach works well if the sensor is not very noisy and the targets are sufficiently apart from each other but does not perform well in cluttered scenarios (Blackman, 2004).

Probabilistic Data Association is the data association technique in which all of the potential candidates for association to a track are combined in a single statistically most probable update, taking account of the statistical distribution of the track errors and clutter and assuming that only one of the candidates is a target, and the rest are false alarms. This technique detailed in (Bar-Shalom and Tse, 1974) has been used by many to solve data association problems. Many different flavors (JPDAF, MXPDAF) of this technique have also been developed, a review of them along with some other techniques have been discussed by Cox (1993) and Bar-Shalom *et al.* (2009). Blackman (2004) has discussed some problems with JPDA (Joint Probabilistic Data Association).

Multiple Hypothesis Tracking (MHT) originally developed by Reid (1979) has been suc-

cessfully used by many tracking systems to solve data association problem like (Burlet *et al.*, 2007, Cox and Hingorani, 1996, Vu *et al.*, 2007). Irrespective of its storage complexity it is a generally preferred method of solving data association problem (Blackman, 2004).

Filtering Techniques

Some most common filtering techniques employed to solve MOT are discussed below.

Kalman Filter developed by Kalman (1960) is by far most commonly used filtering technique in many different fields. In this form this filter only deals with linear systems however its variants have been developed to deal with non linear systems like Extended Kalman Filter, and for highly non linear systems, Unscented Kalman filter.

Particle Filter, Kalman filters are inherently unimodal, so to deal with multi model problems a class of filters commonly known as particle filters were developed (Gordon *et al.*, 1993).

Interacting Multiple Model (IMM) Filter is a preferred filter to track moving objects having multiple motion models (Bar-Shalom and Li, 1995). In IMM a Kalman filter is defined for each motion model. This filter has been extensively used for moving objects tracking (Blackman, 2004, Burlet *et al.*, 2007, Vu *et al.*, 2007).

A detailed derivation of these and many other filters can be found in Thrun *et al.* (2005). Since both *DATMO* and *SLAM* are closely related Wang (2004) has formulated both of them as a single problem called *Simultaneous Localization Mapping and Moving Object Tracking (SLAMMOT)* and has proposed a solution to it.

2.2.3 Synthesis

In this section we have summarized literature review on solutions presented for different parts of the perception problem with some details on modern solutions. We have also listed different techniques developed to address subparts of the main problems of *SLAM* and *DATMO*. Depending on the target environment, a combination of these techniques can be chosen to implement a solution to perception problem. To address this problem for autonomous vehicles driving on busy city roads we have implemented a maximum-likelihood based solution using occupancy grids for *SLAM* using lidar as main sensor. For *DATMO* we have chosen grid based moving object detection along with MHT for data association and IMM for tracking multiple targets. Since we need to detect and track multiple objects moving in different directions with different motion attributes so we need to choose technique which take these requirements into account, with MHT all possible association hypothesis are considered and IMM

can handle multiple motion modes. It makes these techniques quite relevant to our requirements. However low resolution and high uncertainty of the lidar used for this work gives many false alarms and miss detections making it necessary to add another sensor to make for its shortcomings. In next section we discuss perception solutions using multiple sensors.

2.3 Perception with fusion

In previous section we have described some state of the art solutions developed by different people for different parts of the perception problem. A carefully chosen combination of these techniques will give a complete (at least theoretical) solution to the perception problem for any target system. For example for an intelligent vehicle equipped with a laser sensor, choosing following components will give a complete solution to its perception problem:

- Occupancy grid based representation with maximum likelihood SLAM
- Grid based moving objects detection technique
- MHT based data association
- IMM filter for tracking moving objects

Question can be asked, will this be a perfect or even a reliable solution for the target system (autonomous driving on city roads)? A responsible answer will be *'no'*! The reasons being (there may be others):

- Sensor limitations: laser scanners can only get distances of their impact points, hence they see the world as set of points. Even repeated scans can only increase the number of points but will never make the world look continuous. So the state of the world parts between these points can only be guessed. The parts of the world directly observed (the impact points) are just a small fraction of the space in the visible range of the sensor. Similarly some objects are invisible for certain sensors, for example glassy black colors do not reflect enough light to be detected by laser receiver. Similar limitations exist for all sensors.
- The impossibility of modeling the exact behavior of a sensor for all possible situations the target system may encounter while operating. Similarly all factors contributing to the sensor uncertainty are simply never known!

- The strong assumptions and approximations used on virtually every step to facilitate the solution derivation.

Since the first two reasons in the above list may have different significance for different sensors (even for the homogeneous sensors), two systems using same theoretical solution but different sensors may have different results. In the light of this discussion we can say that using multiple sensors may improve perception results because sensors may complement each other with one sensor providing information about an aspect of the environment which is more uncertain for other sensor and vice versa (heterogeneous sensors case). Or even by providing more data about the environment to decrease the part of the environment to be guessed (homogeneous sensors case).

However, using multiple sensors inherently requires us to develop techniques to combine data from these sensors in a way such that the overall results of the system are better than the single sensor case. Since a solution to perception problem consists of multiple steps so this sensor data fusion can also take place on different steps or levels. In this thesis we have developed fusion techniques for laser and vision (stereo camera and mono camera), for three different levels of an intelligent vehicle perception. These levels, already mentioned in first chapter, include: *Low level*; before the input to *SLAM* module, *Object detection level*; before *DATMO* part and *Track level*; for the output of the *DATMO* module (a detailed description of these levels is given in next chapter).

In the following we first give general state of the art on multi-sensor data fusion and then detail the three levels of fusion and give a comparison with works related to our contributions.

2.3.1 Multi-sensor Data Fusion

Multisensor data fusion has been a topic of research since long, the reason being the need to combine information from a number of different sensors to obtain a full and complete description of the environment (Durrant-Whyte *et al.*, 1990). Theoretical scientific literature on data fusion started to appear in late 1960s with implementations and more algorithmic developments in 1970s and 1980s (Gros, 1996). In the mid-1980s (1984), the Joint Directors of Laboratories formed the Data Fusion Subpanel (which later became known as the Data Fusion Group)[1] charged with tasks such as: organizing conferences, promoting data fusion to industry and educational establishments, identifying the needs of industry and coordinating data fusion projects. Since then huge amount of research has been conducted in this field with many researchers, all

[1] `http://en.wikipedia.org/wiki/Data_fusion`

over the world, still dedicating their research efforts to this field broadening the limits of its applicability. Hall and Garga (1999) have highlighted the pitfalls in data fusion that justify the active research still going on. Especially the point *There is no such thing as a magic or golden data fusion algorithm* implies that fusion techniques very much depend on the sensors used, the problem to be solved and even the level of pre-processing done on the sensor data.

Early systems very much focused on integrating information obtained from different positions of a single sensor or multiple sensors to reconstruct the environment. For example Durrant-Whyte (1985) developed a Bayesian integration technique to combine measurement information from distributed sensors. This work (Durrant-Whyte, 1985) is also credited to have started the statistical uncertainty modeling for multi-sensor systems (Nakamura and Zu, 1989). This technique was further used in (Durrant-Whyte, 1986) to fuse observed geometrical features to construct the environment model. Similarly Ayache and Faugeras (1988) have used a Kalman filtering technique to fuse geometrical features extracted from passive vision to reconstruct three dimensional robot environment. In this context Nakamura and Zu (1989) have proposed a general statistical fusion method for multiple sensors based on geometrical uncertainties. In 1987, Luo *et al.* (1987) defined the confidence distance measure using linear Gaussian models and proposed a hypothesis test to reject sensory data obtained by malfunction and discussed an iterative computational method of fusion. Hashimoto and Paul (1987) applied the approach by Durrant-Whyte to integrate encoder data and tachometer data to control manipulators. Yi *et al.* (2000b) have used Dempster-Shafer fusion technique to handle range errors in specular environments of an ultra sonic sensor while building an environment map. Similarly a wealth of publications are available on data fusion for target tracking (Hall and Llinas, 1997). For example Blair and Kazakos (1993) has developed fusion techniques for intermittent sensors for tracking moving targets.

2.3.2 Fusion Levels

The architecture of fusion levels that we have addressed in this dissertation are shown in the figure 2.5.

This architecture divides the robot sensor data processing into four steps with following four levels of fusion:

- *Low level* with minimum preprocessing (like projecting data into a common frame of reference or classifying individual pixels etc) data from each sensor is represented in compatible forms and these individual representation forms are com-

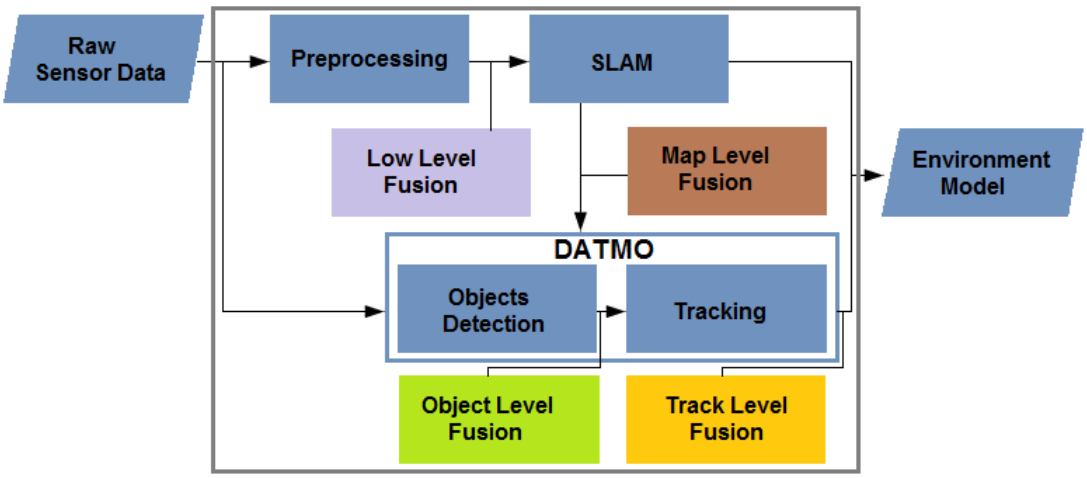

Figure 2.5: *SLAM* and *DATMO* based perception architecture with identified fusion levels.

bined to get fused representation. Output of fusion at this level is input to the *SLAM* module. We have performed fusion at this level between laser and mono vision and laser and stereo vision.

- *Map level*, output of each sensor is processed to solve *SLAM* problem and the maps generated by individual sensors are combined to get a fused map. This level is more interesting for robots moving in cluttered environments, however for intelligent vehicles moving on roads map generated by laser scanner is sufficient so we have not considered this level in our work as explained in next chapter.

- *Object detection level*, output of each sensor is processed to detect objects in the environment, then we perform fusion between these lists of objects to get a list of fused objects, fused objects have more complete properties than their pre fusion versions. Fusion at this level is done between laser and mono vision and laser and stereo vision.

- *Track level*, fusion at this level involves combining lists of moving objects detected and tracked over time by individual sensors. Objective of this fusion is to reduce the number of false tracks. We have performed this fusion between laser and stereo vision.

A practical system may implement fusion on one or more of these levels at the same time. Since laser and vision are complementary sensors so it is interesting to know how information from both of them can be combined in different ways and on different levels of abstraction. Moreover comparing the results and pros and cons of these levels

may facilitate to decide how these information can be optimally used in the context of fast moving intelligent vehicles. Although several other similar works exist however they mostly deal with pedestrians detection and tracking in restricted environments. We do not know any other work treating the same topic in the context of *SLAM* and *DATMO* based intelligent vehicle perception.

In the following we will specify in detail the position of these fusion levels in the usual *SLAM* and *DATMO* based perception setup as shown in figure 2.6. At the end of each level's description we compare our work with related works already done by other researchers.

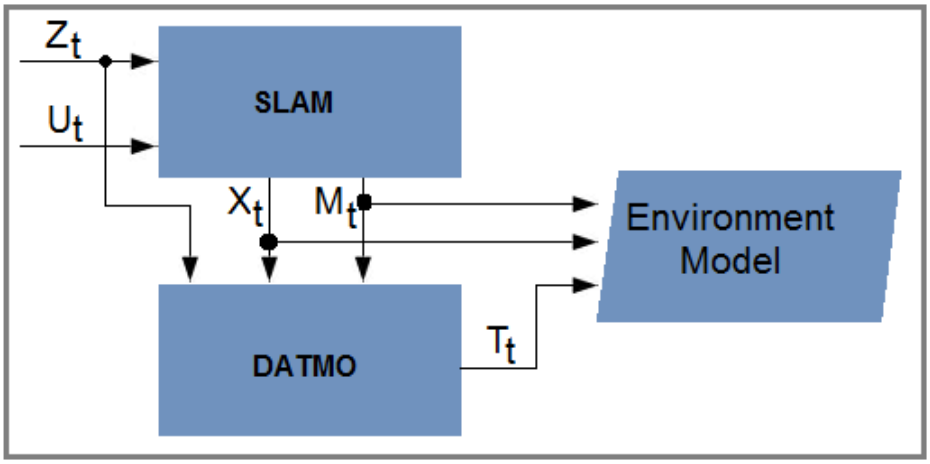

Figure 2.6: Relation of *SLAM* and *DATMO* with their inputs and outputs: Z = sensor input, U = motion measurements (odometry measurements), X = vehicle state, M = generated environment map (relationship of static objects in the environment) and T = tracks (moving objects with motion attributes).

Low level fusion

Position of low level fusion in perception architecture is shown in figure 2.7

We call it low level because only basic preprocessing is done before the fusion process. We have performed fusion at this level between laser and stereo vision and laser and mono vision. Generally fusion at this level involves representing preprocessed data from the sensors in a common format (for example occupancy grids) and then finding some strategy to combine these individual representations to form a combined representation. This combined representation is used as input for later steps.

As it can be seen from the figure 2.7 fusion at this level only affects the *SLAM* part of the perception architecture. More formally the *SLAM* formulation using inputs from

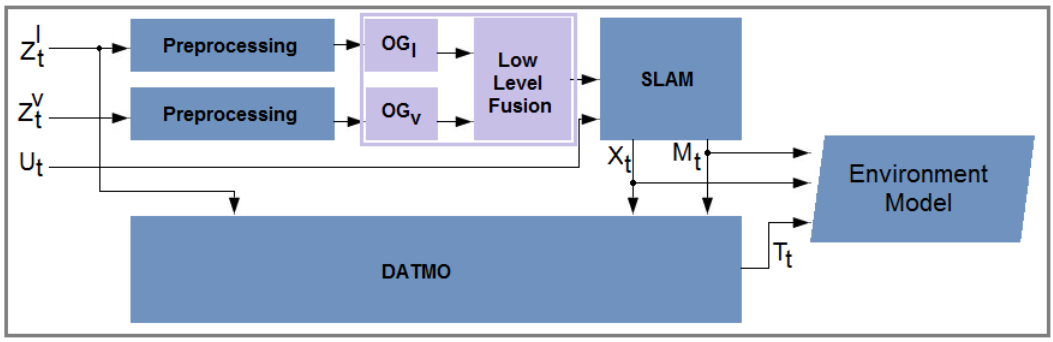

Figure 2.7: Low level fusion with perception architecture

two sensors will be:

$$P(x_t, M|Z_t^l, Z_t^v, U_t)$$

This will require a new derivation of the *SLAM* solution presented in previous section. However with this low level fusion architecture we define following fusion function to combine the two sensor inputs like:

$$Z_t^f = \mathrm{F_{OG}}(Z_t^l, Z_t^v) \tag{2.3.1}$$

And this fused input is used for *SLAM* as follows:

$$P(x_t, M|Z_t^f, U_t)$$

The main purpose of fusion at this level is to compensate the visibility problems of one sensor with the help of second sensor. For example a big truck may not be detected by a laser mounted on a car simply because the level of laser may be lower than the bottom of the truck, however vision sensor will be able to detect it.

In literature we see that Braillon *et al.* (2006) have described an occupancy grid based fusion between stereo and optical flow data. The idea is based on constructing an occupancy grid from stereo data after removing 3D points belonging to ground plane and another grid from optical flow information of two consecutive frames of one camera. This method can be compared with our low level fusion between laser and mono vision. The main difference between the two techniques is that we use a very fast method to calculate occupancy grids for camera (to limit the obstacle search region in the image we use transformations from laser to image reference frames to get a strip like region in the image). Whereas they first calculate the optical flow using complex calculations for each pixel in the image (using two consecutive images) and then using camera projective model they construct an occupancy grid. Moreover while concluding their work they mention some ambiguities for mono camera case which implies that this method had some failure cases.

In our work we have devised a fast method of constructing occupancy grids for vision sensors and then using a weighted approach we combine them with laser data. We have detailed this fusion between mono vision and laser scanner in our publication (Baig and Aycard, 2010). We will explain the construction of occupancy grid for each sensor and the fusion function F_{OG} in chapters 3 and 4 for stereo and mono vision respectively with lidar sensor.

Map level fusion

Next two fusion levels deal with fusion between moving objects detected by both sensors. Another level of fusion can also be defined that fuses the static parts (i.e. map) of the environment detected by multiple sensors. Input for this fusion level will be the output maps of *SLAM* or static objects by these sensors and output will be a combined map. The systems dealing with finding position of the vehicle in the map are also placed in this category. For example work by Abuhadrous *et al.* (2003) deals with vehicle localization on maps and Yi *et al.* (2000a) developed a fusion technique for building precise maps using multiple ultrasonic sensors. Chanier *et al.* (2008) have developed a technique to fuse an on-line map, built using EKF based *SLAM* , and an a priori map to find accurate vehicle pose and to constrain the on-line map using known map.

In this thesis we do not take this fusion level into account because it is more relevant for robots moving in a cluttered environment where both static and dynamic objects can be present and detection of both types of objects is equally important. In the context of intelligent vehicles we suppose that they only move on roads having no permanent static objects on them. To deal with temporary static objects we see that the map generated by laser scanner gives sufficient information about the static parts. So in this context dealing with moving objects becomes more important.

Object level fusion

Object level fusion architecture is shown in figure 2.8. The position of this fusion level is between moving object detection and tracking parts of the *DATMO* .

From the figure we can see that stereo vision processing does follow the same processing steps as that of laser, this is due to the fact that stereo vision processing was done by another team (as already mentioned). Output of their processing consists of a set of detected objects.

Fusion at this level takes two lists of objects, finds corresponding objects in the two lists and fuses their properties. The major challenge here is to solve this correspondence or

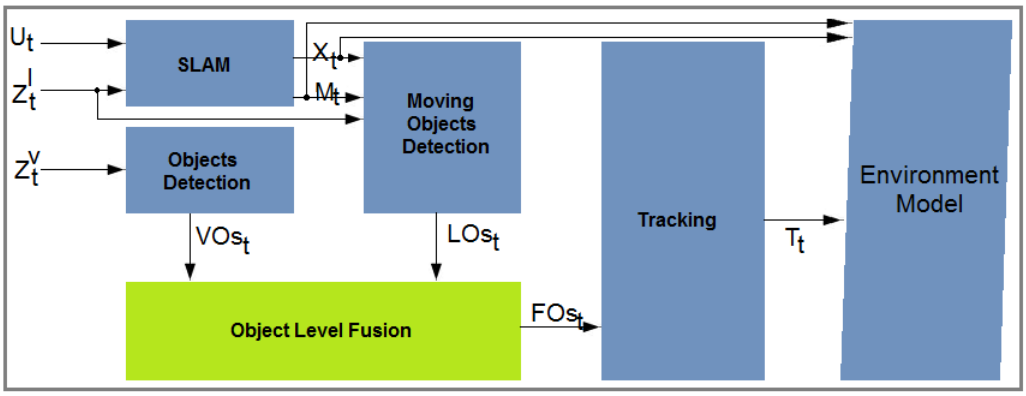

Figure 2.8: Object level fusion with perception architecture. Here VOs_t is the list of objects detected by vision, LOs_t is the list of moving objects detected by laser and FOs_t the list of fused objects.

association problem. This fusion is useful because different sensors detect different properties of the objects more reliably so the fused object has more details than its previous instance. This fusion can be represented in following functional form.

$$FOs_t = \mathrm{F_{OL}}(VOs_t, LOs_t) \tag{2.3.2}$$

Labayrade *et al.* (2005) have presented a fusion technique between laser and stereo vision for obstacles detection. Based on vision technique they extract obstacles from stereo vision data. Using a segmentation technique they perform clustering on laser data to form objects. The fusion mainly consists of comparing the position of objects from both sensors, if an object detected by stereo is on the same position as the one detected by laser then this object is considered a real one. The objects having no matching counterparts in the other sensor are taken as false alarms and are ignored. However we think that the strong assumption about the same position of an object for both sensors (having different uncertainties) is not correct and many of the true objects will be ignored because their position difference for two sensors will make them false alarms.We suggest that fusion at this level should be taken more like an association problem and uncertainties of both sensors should be taken into account during this process.

Somewhat related work in this context has also been done by Baltzakis *et al.* (2003). They have developed a fusion technique between laser and stereo vision data. From laser data they extract line features and each extracted line is supposed to be a wall. Then they construct a 3D model of walls (having infinite height) from all the extracted lines, the walls are supposed to have a fixed color. For each pixel in the stereo camera images they calculate its depth information. Then using projection of a pixel in the image on the constructed 3D model from the laser the color information from this model

are compared with pixel color. If they match then it is inferred that both camera and laser are seeing the same point otherwise camera is seeing a hanging obstacle in the environment. The main disadvantage of this method is its computation cost because for each extracted line it needs to construct a 3D model of the walls, moreover this method is not applicable for outdoor environments where such strong assumptions on shapes and colors cannot be made.

Another important related work is done by Fayad and Cherfaoui (2009). This work based on transferable belief model framework is in-fact a mixture of object and track level fusion as compared to our level scheme. Although this work only addresses the pedestrians, however evidential theory (on which this work is based) is an alternative approach to the Bayesian approach that we have used.

The advantage of fusion at this level is that with more properties of objects we can improve tracking. For example the classification information obtained from vision allow us to apply models on objects to do model based tracking to get smooth tracks. Baig *et al.* (2011) details our work on this level of fusion between stereo vision and laser scanner. We have also performed this fusion between laser and mono vision. The details on how we define the association function are given in chapters 3 and 4 for stereo and mono vision respectively.

Track level fusion

Track level fusion architecture is shown in figure 2.9.

Here again the processing of stereo vision does not perform *SLAM* step due to the fact that we did not do stereo vision processing on raw images.

Fusion at this level requires that input of each sensor is processed to solve *DATMO* problem and for each sensor list of tracks at time t is available. Then the fusion process fuses these lists of tracks to get a combined list of tracks. The fusion function at this level has not only to solve the association problem but also need to find a strategy to fuse their properties (like velocity, position) in a reliable and acceptable way. The fusion function at this level can be given as:

$$T_t = \mathrm{F_T}(T_t^l, T_t^v) \tag{2.3.3}$$

Gidel *et al.* (2009) have developed a particle filter based track level fusion architecture for pedestrian tracking. The main contribution consists of the development of a non parametric data association technique based on machine learning kernel methods. The method described is for pedestrians only.

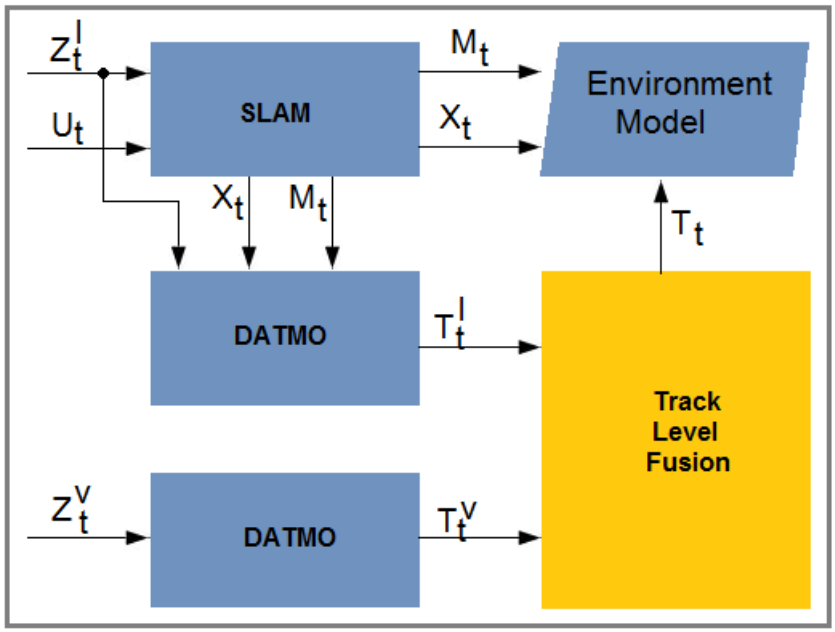

Figure 2.9: Track level fusion with perception architecture. Here T_t^l is the list of tracks detected by laser, T_t^v is the list of tracks detected by stereo vision and T_t the list of fused tracks.

Blanc *et al.* (2004) have described a track level fusion architecture for radar, infrared and lidar sensors. The main concept consists of defining an association technique based on the gating window obtained from covariance matrices of the tracks given by these sensors. This work is very close to what we have done, however we define a simple association function that can be evaluated quickly to validate true tracks. A similar approach has also been described by Floudas *et al.* (2007) using a different association technique.

Fusion at this level is useful for sensors with high noise level. By using an effective fusion strategy false tracks can be reduced. We have performed fusion at this level between laser and stereo vision. Inputs from both sensors are processed separately to detect and track moving objects. Then we define an association function to find corresponding tracks in two lists, details are given in next chapter.

2.4 Road border detection to improve *DATMO*

Our work with different sensors especially laser scanners has shown that most of the false positives (false moving objects) are detected close to the road borders because of low resolution and lidar noise. So to decrease the number of these false alarms an idea is to detect road borders from laser data and then use this border information to find

and remove the moving objects detected close to it. The position of this road border detection module in perception architecture is shown in figure 2.10.

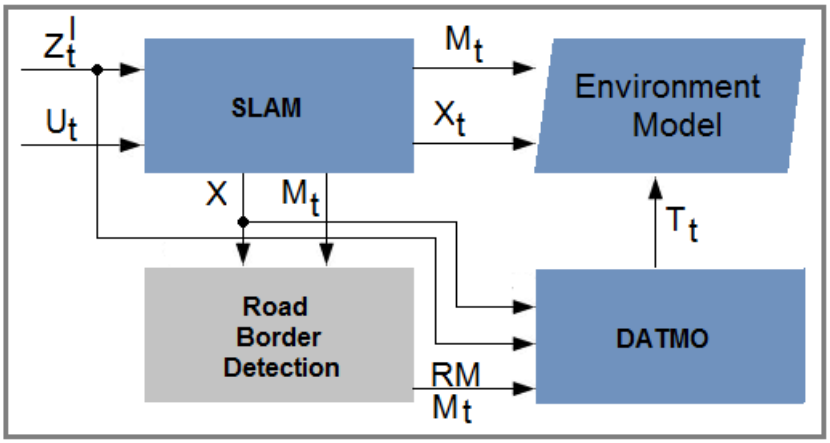

Figure 2.10: Perception architecture with road border detection from the map to improve *DATMO* . *RM* is the road model detected by the road border detection module.

The main idea is to detect road borders efficiently from the map, these borders define road model. This road model is further used during classification of laser hits in dynamic and static segments during *DATMO*. The dynamic segments detected close to the road borders are considered as false alarms and are ignored during further processing of tracking, resulting in reduced number of false tracks. However, moving objects giving consistent tracking results on these areas are not ignored because they may belong to real moving objects like pedestrians walking on side walks.

Many people have already worked on road border detection, especially in vision, lane detection has been a hot field of research. However vision techniques usually based on features extraction are slow. Kirchner and Heinrich (1998) have mentioned pros and cons of usual vision techniques used for this purpose. Moreover using laser scanner they have presented a model based technique for detecting and tracking road borders. Based on clothoid model and Kalman filter it involves many calculations during each step, moreover this work assumes the absence of moving objects in the environment making it infeasible for our work.

Wijesoma *et al.* (2001) present another method for road border detection based on special laser setup along with strict flat road plan assumptions. In their setup laser is mounted on the vehicle such that its beams hit the road plane at few meters from the vehicle, since road is taken as strictly flat, the laser hit points appear as a line on the ground but this line has sharp edges on the borders due to curbs. The algorithm finds these sharp shifts in the data to detect road borders. Due to the fact that this system re-

quires special orientation of laser scanner and strictly flat road, makes it inappropriate for our purpose.

Work by Weiss *et al.* (2007) is closest to what we have done. However rather than finding road borders they find drivable free space in-front of host vehicle in occupancy grid maps by moving multiple line segments in-front of vehicle in both left and right sides. These line segments stop when they reach occupied cells on both sides and then they extract the polygonal shape from these segments as drivable area. For this technique to work a high resolution accurate laser scanner is required otherwise with noisy data search lines will stop on different positions giving irregular shapes.

An important contribution of our work in this context is related to an efficient road border detection algorithm from laser data and to use it further as an a priori knowledge for *DATMO* module to reduce false tracks due to sensor noise. Based on intermediate grid and noise filtering we define a fast search technique to find road borders in the map generated by *SLAM*. We detail this process in next chapter.

2.5 Summary

In this chapter we have presented formalizations for *SLAM* and have described different solutions presented in the literature to solve it. Among different choices available we have proposed to use occupancy grid based map representation for laser sensor and maximum likelihood based *SLAM* solution. We have also summarized solutions to *DATMO* problem and different solution choices available for its components. For an intelligent vehicle system we propose to use MHT based data association with IMM filtering techniques to handle multiple moving objects tracking, for dynamic objects having different motion modes.

Although these components give a solution to the vehicle perception problem however due to sensor uncertainties there usually are many false positives. In order to deal with them we propose to use vision (stereo and mono) as complementary sensor with laser scanner. However using multiple sensors requires to devise fusion techniques to combine information from these sensors. In this context we propose fusion at three different levels (low, object and track levels) in the *SLAM* and *DATMO* based vehicle perception architecture. We have specified the position, inputs and outputs of these fusion levels in this architecture and have also described some earlier works carried out in this regard with pros and cons.

Finally it has been stated that road border information can be inferred efficiently from

the map constructed with laser data and these information can be used to detect false moving objects in noisy laser data near these borders, to improve perception results.

In next chapters we will expand on these concepts by giving details of preprocessing done for vision (mono and stereo) and laser sensors. We will also formally develop the fusion functions mentioned in this chapter for different fusion levels and will present results of these fusion strategies for data sets obtained from Volkswagen demonstrator vehicle.

CHAPTER 3

Laser and Stereo Vision Fusion

Both laser scanners and vision cameras are sensors of choice for robotic perception these days. With laser providing precise distance information and vision giving content rich information, like, classification information of objects, host vehicle is able to build more elaborate environment model. Decisions based on this model are more useful and intended. Stereo camera in vision sensors is even more useful than mono vision because other than usual classification information, we can also get depth information of objects using epipolar geometry making it compatible with laser sensor and creating some redundancy that may be useful when one sensor fails temporarily.

In this chapter we present our work of fusion on three levels between laser and stereo vision. These fusion levels are defined in the context of *SLAM* and *DATMO* based perception architecture. Our work is an extension of work by Vu (2009) which was based on laser scanner only. Using a different laser scanner with relatively low resolution and more uncertainty a reimplementation of this solution gives many miss detections and false alarms. However these problems can be greatly reduced by using stereo vision along with laser scanner and using an effective fusion strategy at an appropriate level. This chapter deals with this and is structured as follows:

In next section, before going into fusion details, we give details of our occupancy grid based theoretical solution to both *SLAM* and *DATMO* problems. Later sections will give implementation details of these problems. Low level fusion between laser and stereo vision sensors is detailed in section 3.2, especially in this section we explain inverse sensor models for both sensors for constructing occupancy grids. Section 3.3 explains our object level fusion, after explaining the extraction of moving objects from laser and input list of objects from stereo vision we detail the fusion process and present some results. Track level fusion is presented in section 3.4. We detail our road border detection algorithm in section 3.5 and explain how this knowledge can be used to re-

duce false alarms detected near road boundaries. Last section gives a synthesis of this chapter.

3.1 Occupancy grid based SLAM and DATMO

3.1.1 SLAM

In occupancy grid based environment representation the environment is divided into two dimensional lattice M of rectangular cells m_i of equal sizes, where each cell has a probabilistic estimate of its occupancy state. At any instant of time the state of this grid represents the map of the environment. In this section we first present the solution of mapping part assuming that position of the vehicle is known (i.e. x_t is known) and then we will discuss how to find best localization given updated map.

The objective of the mapping algorithm is to estimate the posterior probability $P(M|x_{1:t}, z_{1:t})$ where $x_{1:t} = \{x_1, \ldots, x_t\}$ is the vehicle trajectory and $z_{1:t} = \{z_1, \ldots, z_t\}$ are the sensor inputs obtained from start to time t. Using Bayes theorem this probability can be expanded to a form where we are required to calculate a marginal probability over the map M for each new sensor input making it intractable for even small maps. A proposed solution is to assume conditional independence of the map cells reducing above probability to $P(m|x_{1:t}, z_{1:t})$ where mapping is equivalent to estimating occupancy state of all individual cells. Using Bayes theorem we can write it as:

$$P(m|x_{1:t}, z_{1:t}) = \frac{P(z_t|x_{1:t}, z_{1:t-1}, m).P(m|x_{1:t}, z_{1:t-1})}{P(z_t|x_{1:t}, z_{1:t-1})} \tag{3.1.1}$$

We can further assume that current sensor measurement z_t does not depend on previous measurements ($z_{1:t-1}$) and previous vehicle positions ($x_{1:t-1}$). This reduces the factor $P(z_t|x_{1:t}, z_{1:t-1}, m)$ to $P(z_t|x_t, m)$ so that above equation becomes:

$$P(m|x_{1:t}, z_{1:t}) = \frac{P(z_t|x_t, m).P(m|x_{1:t}, z_{1:t-1})}{P(z_t|x_{1:t}, z_{1:t-1})} \tag{3.1.2}$$

Applying Bayes theorem on the factor $P(z_t|x_t, m)$ this equation becomes:

$$P(m|x_{1:t}, z_{1:t}) = \frac{P(m|z_t, x_t).P(z_t|x_t).P(m|x_{1:t}, z_{1:t-1})}{P(m|x_t).P(z_t|x_{1:t}, z_{1:t-1})} \tag{3.1.3}$$

Here $P(m|x_t)$ is the prior probability of occupancy of the cell m and does not depend on the current vehicle position, so above equation becomes.

$$P(m|x_{1:t}, z_{1:t}) = \frac{P(m|z_t, x_t).P(z_t|x_t).P(m|x_{1:t}, z_{1:t-1})}{P(m).P(z_t|x_{1:t}, z_{1:t-1})} \tag{3.1.4}$$

In this equation the term $P(m|x_{1:t}, z_{1:t-1})$ can be expanded using Bayes theorem as:

$$P(m|x_{1:t}, z_{1:t-1}) = \frac{P(x_t|x_{1:t-1}, m, z_{1:t-1}).P(m|x_{1:t-1}, z_{1:t-1})}{P(x_t|x_{1:t-1}, z_{1:t-1})} \tag{3.1.5}$$

So the equation 3.1.4 becomes

$$P(m|x_{1:t}, z_{1:t}) = \frac{P(m|z_t, x_t).P(z_t|x_t).P(x_t|x_{1:t-1}, m, z_{1:t-1}).P(m|x_{1:t-1}, z_{1:t-1})}{P(m).P(z_t|x_{1:t}, z_{1:t-1}).P(x_t|x_{1:t-1}, z_{1:t-1})} \tag{3.1.6}$$

This equation (3.1.6) gives the probability that cell m is occupied, similarly the probability that cell m is free (represented as $\overline{m}$ such that $P(\overline{m}) = 1 - P(m)$) can be given as:

$$P(\overline{m}|x_{1:t}, z_{1:t}) = \frac{P(\overline{m}|z_t, x_t).P(z_t|x_t).P(x_t|x_{1:t-1}, \overline{m}, z_{1:t-1}).P(\overline{m}|x_{1:t-1}, z_{1:t-1})}{P(\overline{m}).P(z_t|x_{1:t}, z_{1:t-1}).P(x_t|x_{1:t-1}, z_{1:t-1})} \tag{3.1.7}$$

Dividing equation (3.1.6) by equation (3.1.7) we get:

$$\frac{P(m|x_{1:t}, z_{1:t})}{P(\overline{m}|x_{1:t}, z_{1:t})} = \frac{P(m|z_t, x_t)}{P(\overline{m}|z_t, x_t)}.\frac{P(\overline{m})}{P(m)}.\frac{P(x_t|x_{1:t-1}, m, z_{1:t-1})}{P(x_t|x_{1:t-1}, \overline{m}, z_{1:t-1})}.\frac{P(m|x_{1:t-1}, z_{1:t-1})}{P(\overline{m}|x_{1:t-1}, z_{1:t-1})} \tag{3.1.8}$$

Since we suppose that current position x_t is known so the terms of the fraction $\frac{P(x_t|x_{1:t-1}, m, z_{1:t-1})}{P(x_t|x_{1:t-1}, \overline{m}, z_{1:t-1})}$ cancel each other giving following equation:

$$\frac{P(m|x_{1:t}, z_{1:t})}{P(\overline{m}|x_{1:t}, z_{1:t})} = \frac{P(m|z_t, x_t)}{P(\overline{m}|z_t, x_t)}.\frac{P(\overline{m})}{P(m)}.\frac{P(m|x_{1:t-1}, z_{1:t-1})}{P(\overline{m}|x_{1:t-1}, z_{1:t-1})} \tag{3.1.9}$$

If we define:

$$Odds(x) = \frac{P(x)}{P(\overline{x})} = \frac{P(x)}{1 - P(x)} \tag{3.1.10}$$

Then equation (3.1.9) becomes:

$$Odds(m|x_{1:t}, z_{1:t}) = Odds(m|z_t, x_t).Odds(m)^{-1}.Odds(m|x_{1:t}, z_{1:t-1}) \tag{3.1.11}$$

Usually the probability values are very small and division operation makes them even smaller. So to avoid the underflow of values we represent them in logarithmic forms. This can be achieved by applying *log* operation on both sides of the equation (3.1.11), giving following log odds form:

$$\log Odds(m|x_{1:t}, z_{1:t}) = \log Odds(m|z_t, x_t) - \log Odds(m) + \log Odds(m|x_{1:t-1}, z_{1:t-1}) \tag{3.1.12}$$

$P(m)$ and $P(\overline{m})$ are the prior probabilities of cells being occupied and free respectively. We usually take them as $P(m) = P(\overline{m}) = 0.5$ since we have no a priori information about them. Because of being equal they cancel out each other in equation (3.1.9) and equation (3.1.12) reduces to:

$$\log Odds(m|x_{1:t}, z_{1:t}) = \log Odds(m|z_t, x_t) + \log Odds(m|x_{1:t-1}, z_{1:t-1}) \tag{3.1.13}$$

The value of the probability $P(m|x_{1:t}, z_{1:t})$ can be recovered by using the definition of *Odds* operator in equation (3.1.11) and doing some simplifications, this gives us:

$$P(m|x_{1:t}, z_{1:t}) = \left[1 + \frac{1 - P(m|x_t, z_t)}{P(m|x_t, z_t)} . \frac{P(m)}{1 - P(m)} . \frac{1 - P(m|x_{1:t-1}, z_{1:t-1})}{P(m|x_{1:t-1}, z_{1:t-1})}\right]^{-1} \tag{3.1.14}$$

From this equation (3.1.14) it is clear that to update the occupancy state of a cell from new measurements we need to multiply previous odds ratio of its occupancy state with odds ratio of current belief and divide by prior odds ratio. Until here these derivations are generic and can be used for any sensor, however in later sections we will see how to calculate the current belief (also called inverse sensor model) term ($P(m|x_t, z_t)$) for each sensor.

For above derivations we had supposed that current position (x_t) of the vehicle is known, however for *SLAM* we have to estimate it simultaneously from the map and sensor input. As described in chapter 2, we have used an incremental maximum likelihood *SLAM* approach. According to this approach the current vehicle pose estimate $\hat{x}_t$ can be calculated by maximizing the marginal likelihood given as:

$$\hat{x}_t = \arg\max_{x_t} \{P(z_t|x_t, \hat{M}_{t-1}) P(x_t|u_t, \hat{x}_{t-1})\} \tag{3.1.15}$$

Where $\hat{M}_{t-1}$ is the map at time instant $t-1$ and $\hat{x}_{t-1}$ is the estimated previous pose. Below we will explain how to maximize the marginal likelihood. Having determined the pose of vehicle we update the occupancy grid map using previous map and new measurement as given in the equation (3.1.14). This update process gives new map and can be represented as:

$$\hat{M}_t = \hat{M}_{t-1} \bigcup \overline{M} \tag{3.1.16}$$

Where $\overline{M}$ is the new instantaneous map obtained from estimated pose $\hat{x}_t$ and current measurement z_t.

The two probabilities given in equation (3.1.15) are important to consider. The term $P(x_t|u_t, \hat{x}_{t-1})$, called motion model, gives the probability of being at position x_t when control u_t is applied from previous estimated position $\hat{x}_{t-1}$. And $P(z_t|x_t, \hat{M}_{t-1})$ is the measurement model and gives the probability of getting the measurement z_t from pose x_t and previous estimated map $\hat{M}_{t-1}$. Now maximizing (3.1.15) is equivalent to finding the vehicle pose x_t that satisfies the motion model $P(x_t|u_t, \hat{x}_{t-1})$ and best fits the z_t to the map $\hat{M}_{t-1}$. In this setup the maximization problem reduces to defining a probabilistic motion model, sampling sufficient values of new pose x_t from this model and for each value fitting the new measurement z_t to the map $\hat{M}_{t-1}$. The sample of x_t that

best fits (a score function can be defined to decide the best fit) z_t can be taken as the new estimated pose $\hat{x}_t$. For a control input consisting of translational and rotational velocities $u_t = (v_t, \omega_t)$ a probabilistic motion model is shown in figure 3.1.

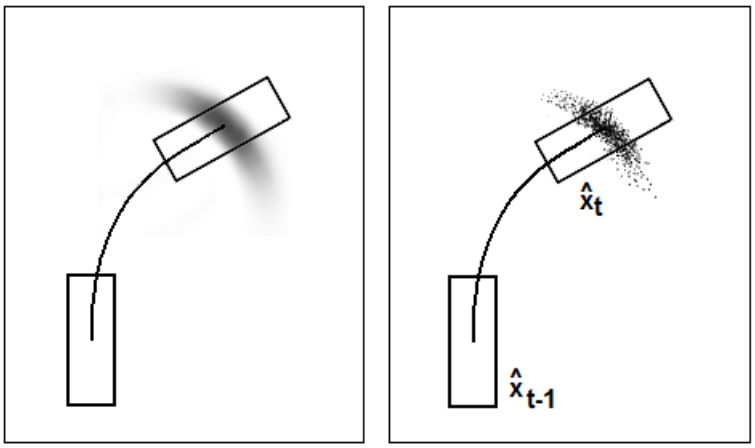

Figure 3.1: Vehicle motion model $P(x_t|u_t, \hat{x}_{t-1})$ (left) and its sampling version (right). Gray part shows the uncertainty in the two components of control input $u_t = (v_t, \omega_t)$.

The probability $P(z_t|x_t, \hat{M}_{t-1})$ can be taken equivalent to the degree the measurement z_t fits to the map $\hat{M}_{t-1}$ w.r.t the current sampled value of x_t. For example to evaluate this probability for a laser scanner we project the current measurement z_t on the map constructed so far $\hat{M}_{t-1}$ taking current sampled value of x_t as position of the vehicle. If c_i is the grid cell corresponding to the hit point of ith laser beam in z_t and $\hat{M}_{t-1}[c_i]$ is the occupancy probability of cell c_i in the map $\hat{M}_{t-1}$ then the probability $P(z_t|x_t, \hat{M}_{t-1})$ is given as:

$$P(z_t|x_t, \hat{M}_{t-1}) \propto \sum_{i}^{N} \hat{M}_{t-1}[c_i] \tag{3.1.17}$$

Here N is the total number of laser beams in z_t. This sum can be evaluated for different samples of x_t, and the one giving maximum value of this sum will be the current estimated pose $\hat{x}_t$ according to the equation (3.1.15). This is called scan matching *SLAM* approach for maximum likelihood based *SLAM* solution.

Summarizing we can say that maximum likelihood based *SLAM* consists of finding best vehicle pose estimate $\hat{x}_t$ according to the equation (3.1.15) using motion model $P(x_t|u_t, \hat{x}_{t-1})$ and measurement model $P(z_t|x_t, \hat{M}_{t-1})$ based on scan matching approach. And then using this pose $\hat{x}_t$ and the latest sensor measurement z_t to update the map as given in equation (3.1.16) using the cell update scheme given by equation (3.1.14).

In next section we discuss the low level fusion scheme that we have developed for laser and stereo vision sensors.

3.1.2 DATMO

The quality of the map and the precision of the localization mentioned in previous section are affected if the measurements belonging to dynamic objects in the environment are not distinguished from the measurements given by static parts. To get good results, input to *SLAM* should only consist of measurements belonging to static parts of the environment, so it is important to detect moving objects in the environment. To do this we make use of inconsistencies observed between new data and the map constructed so far. From the updated position of the vehicle new sensor measurements are projected on the map. Measurements falling in the cells that were seen as free previously are considered as belonging to dynamic objects. If $M[z_t^i]$ (here M=$\hat{M}$, the estimated map) is the cell where ith part of measurement z at time t is projected (like ith laser beam of laser scan z_t) then the dynamic parts of the measurements is the set $\{z_t^i | M[z_t^i]$ is marked free$\}$. This gives a theoretical solution based on scan matching technique to the moving objects detection problem.

After detection of moving objects next step is to track them, where tracking is essentially identifying same moving object in consecutive data frames and calculating its properties like: position, velocity, acceleration etc. Since number of moving objects visible to the sensors of the vehicle may change from one data frame to next and due to different modes and directions of object movements, tracking usually is a non trivial task. Theoretically if ω is any set of tracks at time t then tracking is the optimal set of tracks $\omega^* = \underset{\omega}{\text{argmax}} P(\omega | z_{1:t})$. We will explain the components of the tracking solution in 3.3.4 section.

These two subsections complete the theoratical solution to *SLAM* and *DATMO* problems. Implementation details of mapping part of *SLAM* problem are given in next section whereas solution to localization part is given in section 3.3.1 along with solution to *DATMO* problem using lidar and stereo vision sensors.

3.2 Low level fusion

Position of low level fusion in the perception architecture is shown in figure 3.2. Fusion at this level consists of processing sensor data (latest scan) to construct occupancy grid for each sensor separately and then combining these grids to get a fused grid. Since we are using laser and stereo vision at this level, we represent their current data scans as z_t^l and z_t^v respectively. In the following we explain how we construct occupancy grids from these data and then how we fuse them to get a fused grid. This fused grid is then

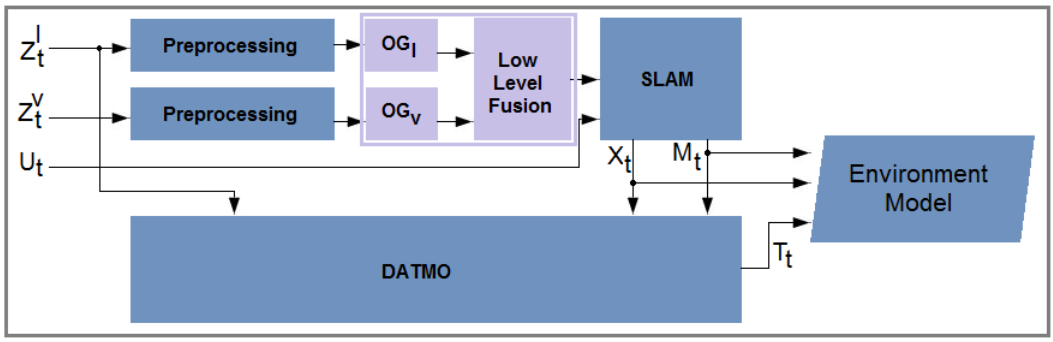

Figure 3.2: Low level fusion with perception architecture

input to the *SLAM* module to update the grid map $\hat{M}_{t-1}$ to get $\hat{M}_t$

3.2.1 Laser processing: occupancy grid construction

In this section we explain how we build occupancy grid from current laser scan z_t^l. Laser scan consists of N tuples $\{(d_1, \theta_1), \ldots, (d_N, \theta_N)\}$ corresponding to its N laser beams' hit points (usually a single laser beam is reflected by a rotating mirror at N different angles to get N distance readings). Each laser beam gives the angle (θ) and distance (d) of the point where it impacts on the object in its way.

An occupancy grid (OG_l) having the same dimensions and cell sizes as that of grid map used by *SLAM*, is created. The current pose estimate $\hat{x}_t$ is taken as the position of the vehicle in this grid. Figure 3.3 shows an illustration with vehicle (blue rectangle) in an occupancy grid, laser beams (red lines), laser field of view (black closed shape) and an example cell $OG_l[i][j]$, named m, of which we need to estimate occupancy state or more precisely the probability $P(m|\hat{x}_t, z_t^l)$.

If θ_1 and θ_N are the angles of first and last laser beams respectively and d_{max}^l is the maximum distance laser scanner can measure then estimating the inverse sensor model probability ($P(m|\hat{x}_t, z_t^l)$) for each grid cell involves following steps:

1. Calculating the polar coordinates (d_m, θ_m) of the center of the cell m w.r.t the laser frame of reference. It means that the length and angle of blue line w.r.t the origin of green frame of reference in the figure 3.3.

2. Finding the closest laser beam (d_k, θ_k) by comparing θ_m with laser beam angles θ_i. This closeness only depends on the proximity of the angles and does not depend on the beam length. Ideally a beam should only be used to estimate the occupancy state of the cells lying on the way of the laser beam (with $\theta_m = \theta_k$), however, since the number of beams is limited we suppose that the cells lying

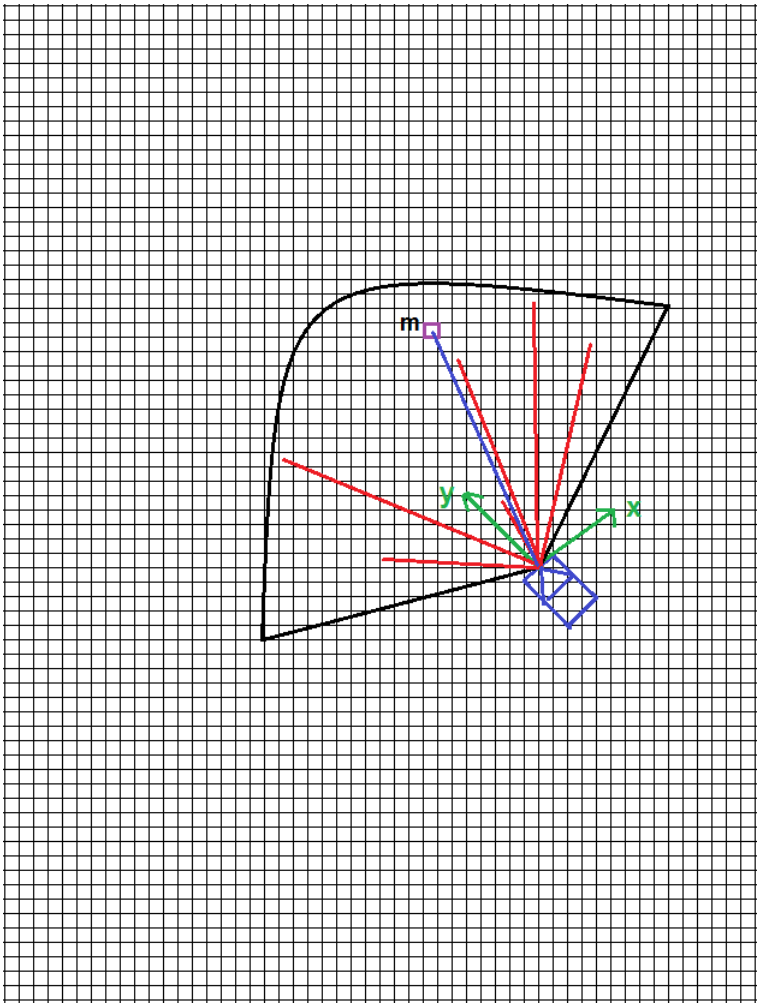

Figure 3.3: Illustration showing current position of the vehicle (blue) in the occupancy grid OG_l, laser frame of reference (green), laser beams (red), laser field of view (black) and an example cell $m = OG_l[i][j]$ of which we want to estimate the occupancy state.

close to the closest beam have the same occupancy state as the ones lying on its path.

3. Calculating the cell occupancy probability using following equation:

$$P(m|\hat{x}_t, z_t^l) = \begin{cases} p_{prior} & \text{if } \theta_1 > \theta_m > \theta_N \text{ or } d_m > d_{max}^l \\ p_{free} & \text{if } d_m < d_k \\ p_{occ} & \text{if } d_m = d_k \\ p_{prior} & \text{otherwise} \end{cases} \tag{3.2.1}$$

Here p_{prior} is the prior probability, p_{free} is the probability that the cell is free and p_{occ} is the value of probability when the cell is occupied. From the inverse sensor model shown in figure 3.4 they are respectively taken as 0.5, 0.2 and 0.8.

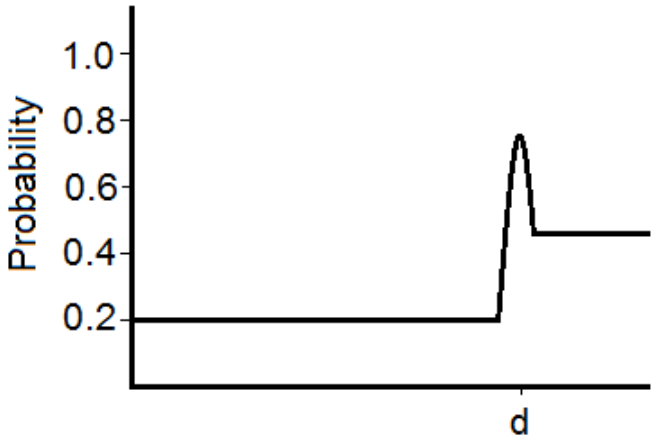

Figure 3.4: Laser scanner inverse sensor model.

Evaluating above three steps for each cell of the grid gives the occupancy grid for z_t^l. However the area visible in the laser field of view is relatively small than the occupancy grid size, so to quickly calculate the occupancy state of each cell we first initialize all cells with p_{prior} value then we check if the current cell lies in the laser field of view (the condition: $\theta_1 > \theta_m > \theta_N$ or $d_m > d_{max}$), the cells passing this condition are used for occupancy calculations. An occupancy grid for real laser data constructed using above steps is shown in the figure 3.5.

3.2.2 Stereo vision processing

Stereo vision processing (done by: Nedevschi *et al.* (2009a,b)) for this level of fusion consists of taking the two images from the stereo camera, rectifying them, correcting them (to remove lens distortions) and down sampling them. The down sampled images are input to the stereo reconstruction module implemented in the hardware. From the depth map provided by the stereo-engine the 3D coordinates of reconstructed points are computed using camera parameters. The output of this processing consists of a list of 3D points ($z_t^v = \{P_1, \ldots, P_M\}$ where $P_i = (x_i, y_i, z_i)$ and M is points count) and our objective in this section is to build an occupancy grid from these points that should be compatible with the one constructed from laser data. The 3D coordinates of these points are given w.r.t a frame of reference having origin on ground exactly under the laser scanner with z-axis pointing in the direction of driving and x-axis to the right, moreover if ground is flat then x-z and ground planes coincide.

The laser plane is at a height of $h = 33cm$ from this x-z plane (laser scanner is mounted in the front bumper of the vehicle at this height from the ground). Since 3D stereo

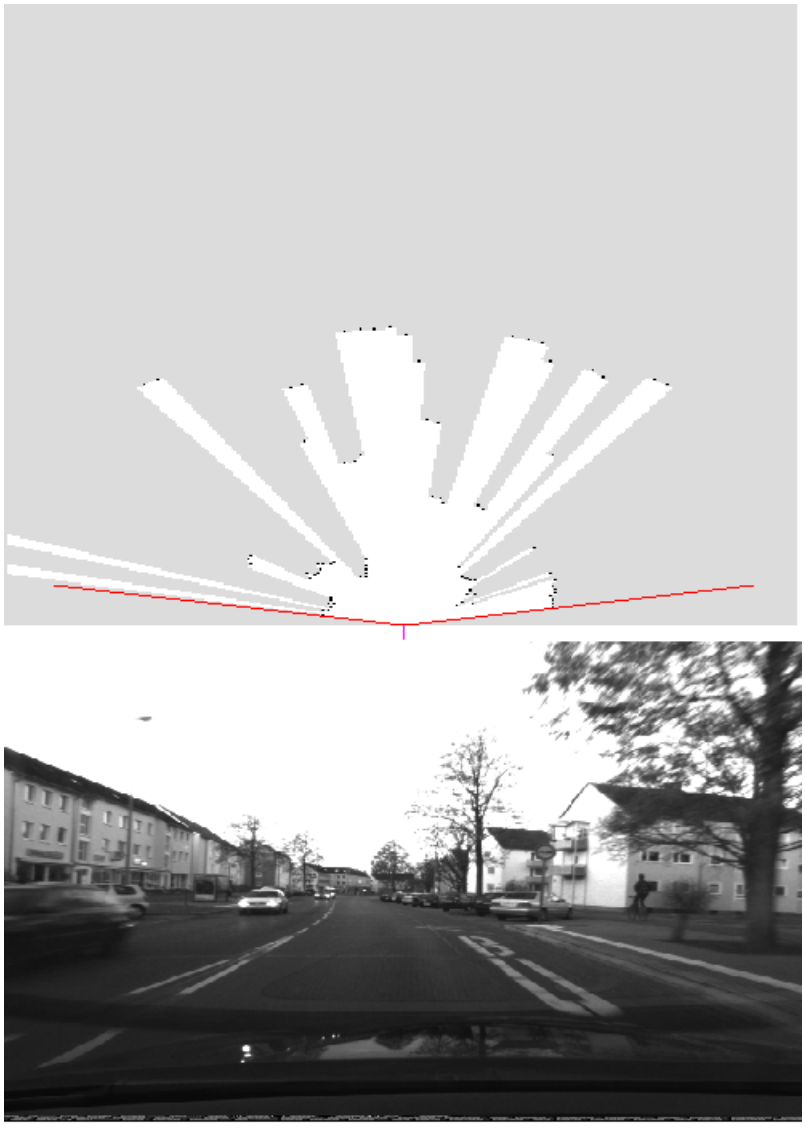

Figure 3.5: An occupancy grid constructed from laser data for the road scenario shown at the bottom. Red lines mark the field of view of laser scanner.

reconstruction gives points everywhere (for example ground points, tree leaves, obstacles etc) so to construct an occupancy grid compatible with the one constructed from the laser data, we need to find vision points which most probably belong to the obstacles detected by laser. To meet this requirement we ignore the points which are below the level of laser plane and more than one meter above this level. This gives us following reduced set of points:

$$\overline{z}_t^v = \{P_i \,|\, P_i \in z_t^v \,\&\, h > y_i > h+1\} \tag{3.2.2}$$

This set of points corresponds to a layer of points of 1 meter thickness at a height of h from the ground. Figure 3.6 shows an example layer of points (from two different angles) belonging to the objects shown in figure 3.6c. For next step of preprocessing

we define an intermediate grid G_c (grid of counts) having same dimensions, cell sizes and position as that of the laser grid (the position of this grid is on the bottom of this layer of points, same as that of laser plane) and project all points on this grid (figure 3.7 a). After projection each cell has a count of the points falling in it (projection simply involves ignoring the y coordinate and determining the grid cell exactly below the point and incrementing the count of the cell for each point falling in it, the count of the cell is zero if no point falls in it) we need this point density for each cell to remove noise.

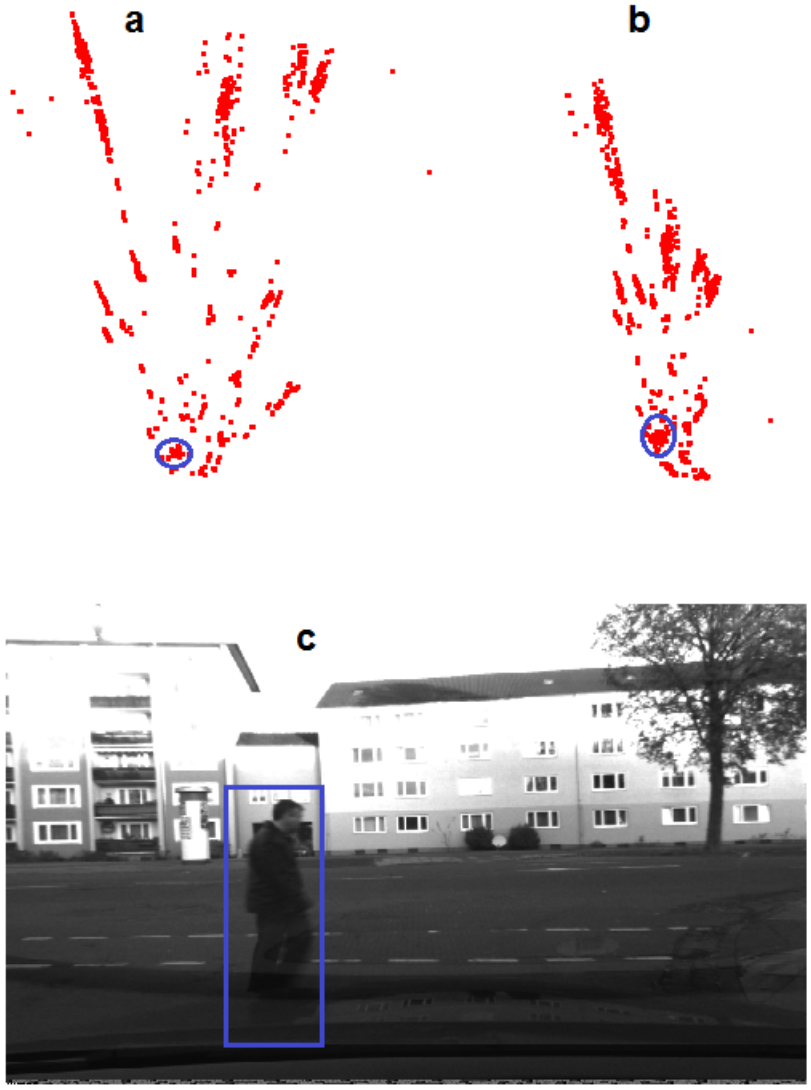

Figure 3.6: Layer of points (a) for a given real scenario (c). Same layer of points (b) rotated about 45° clockwise in 3D. Position of the pedestrian shown in blue circles.

As a noise removal step we define a grid $\overline{G}_c$ similar to G_c and populate it as follows:

$$\overline{G}_c[i][j] = \begin{cases} 1 & \text{if } \sum_{k=i-1}^{i+1} \sum_{l=j-1}^{j+1} G_c[k][l] > V_{thr} \\ 0 & \text{otherwise} \end{cases} \tag{3.2.3}$$

Where V_{thr} is the threshold count value, determined empirically, to remove noise. After this step we have a 1 in the cell where the probability of the presence of an object is high. The cluster of the adjacent cells having 1 in them, form an object. The grid G_c before and after noise removal is shown in figure 3.7(a,b) for a road scenario.

The last step of preprocessing defines the inverse sensor model that we use to calculate occupancy probability for each cell. This step consists of finding the cone of each object with the vertex of the cone at the origin and its two sides passing through the two extreme points of the objects determined in last step. To define such a cone for each object we find polar coordinates of each cell of the object, cells having the minimum and maximum angles form the two ends of the object and lines passing through these cells and the origin form the cone of the object (such a cone for an object is shown in figure 3.7 c). In this context suppose that θ^i_{min} is the minimum and θ^i_{max} is the maximum angle of the ith object and r^i_{min} is the distance of this object's cell closest to the origin. Similarly, if θ_s is the vision field of view start angle, θ_e is the vision field of view end angle and d^v_{max} is the stereo vision effective range then we define the inverse sensor model for vision for each cell (having polar coordinates (r_m, θ_m)) of its occupancy grid OG_v as follows:

$$P(m|\hat{x}_t, \overline{z}^v_t) = \begin{cases} p^v_{prior} & \text{if } \theta_s > \theta_m > \theta_e \text{ or } d_m > d^v_{max} \\ p^v_{occ} & \text{if } \overline{G}_c[m] = 1 \\ p^v_{free} & \text{if } \theta^i_{max} >= \theta_m >= \theta^i_{min} \text{ and } r_m <= r^i_{min} \\ p^v_{prior} & \text{if } \theta^i_{max} >= \theta_m >= \theta^i_{min} \text{ and } r_m > r^i_{min} \\ p^v_{free} & \text{otherwise} \end{cases} \tag{3.2.4}$$

In simple words the cells lying between the vertex of the cone and the object are marked as free whereas the cells beyond the object in the cone are marked as unknown. The cells belonging to the object are marked as occupied. Finally the cells which do not lie in the cone of any object are marked as free because no object was found by stereo vision in those parts. These steps of preprocessing and inverse sensor model are shown in figure 3.7.

3.2.3 Fusion

After that two compatible grids have been constructed for both sensors we can combine them using some appropriate scheme that depends on the knowledge about the sensor

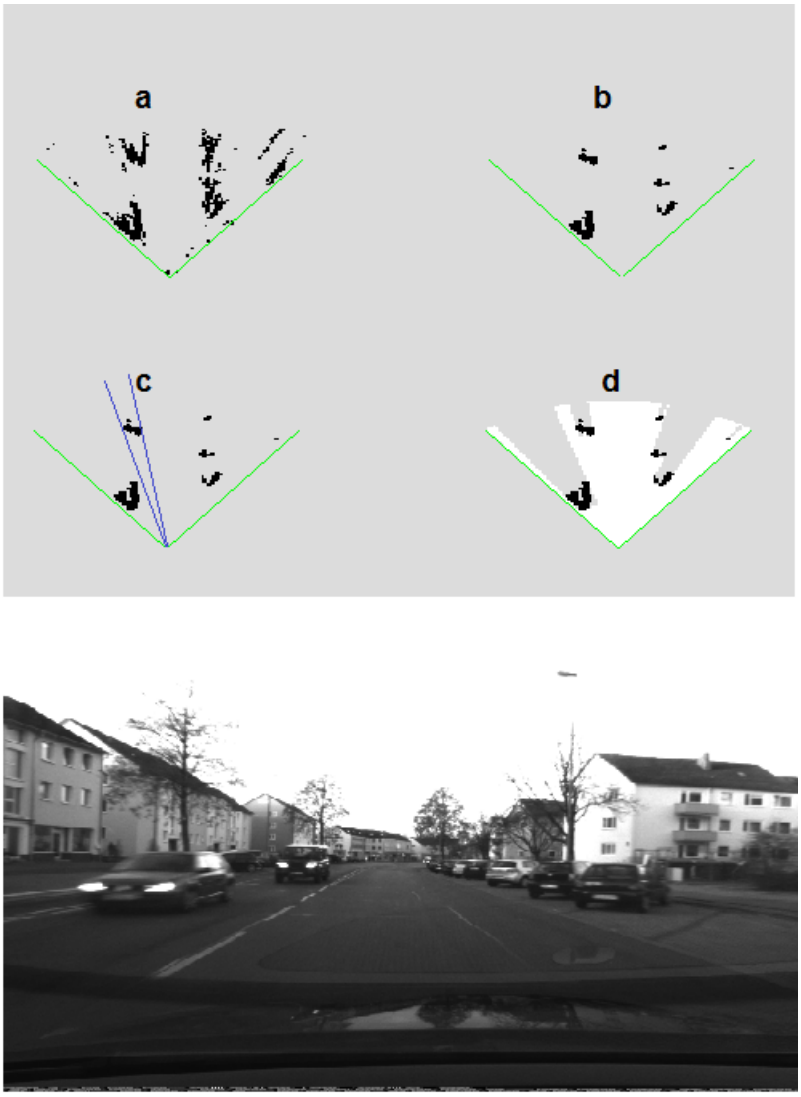

Figure 3.7: Steps to construct an occupancy grid for vision points. a) points projected on a grid, b) object points after noise removal, c) object cone for one object shown by blue lines, d) final occupancy grid for vision points. Image corresponding to these grids shown at the bottom.

behavior.

A general weighted scheme can be given as (α being the weighting factor):

$$OG_f[i][j] = \alpha.OG_l[i][j] + (1-\alpha)OG_v[i][j] \tag{3.2.5}$$

Although this fusion scheme can assign weight according to the uncertainty of the sensor however this scheme does not deal with conflicting situations where one sensor sees an object but the other does not or vice a versa. This scheme is more useful where we want to be sure about the free or occupied state. If both sensors see same state of space then the probability of actual state being the opposite is highly unlikely.

In our case the laser scanner has the problem of getting reflections from the white lane markings and missing thin poles due to weak resolution (2° for data sets having 3D stereo points). Because of these problems we give more weight to the vision grid within its view. We can write fusion function as:

$$OG_f[i][j] = \begin{cases} OG_v[i][j] & \text{for } i, j \text{ in vision field of view} \\ OG_l[i][j] & \text{otherwise} \end{cases} \tag{3.2.6}$$

3.2.4 Results

Some fusion results are shown in figures 3.8 and 3.9. Scenario shown in figure3.8 shows the laser reflections from white lane markings giving false objects (shown in 3.8a) in front of the vehicle, however no such objects are seen by vision (shown in 3.8b) so a fused grid (shown in 3.8c) gives more realistic environment model. Similarly a false object detected by laser has been removed by the fusion in figure 3.9. The main thing in low level fusion is to define appropriate inverse sensor models and then constructing individual grids. Once the grids are ready then an appropriate way of combining these grids can be defined that take the sensor properties into account.

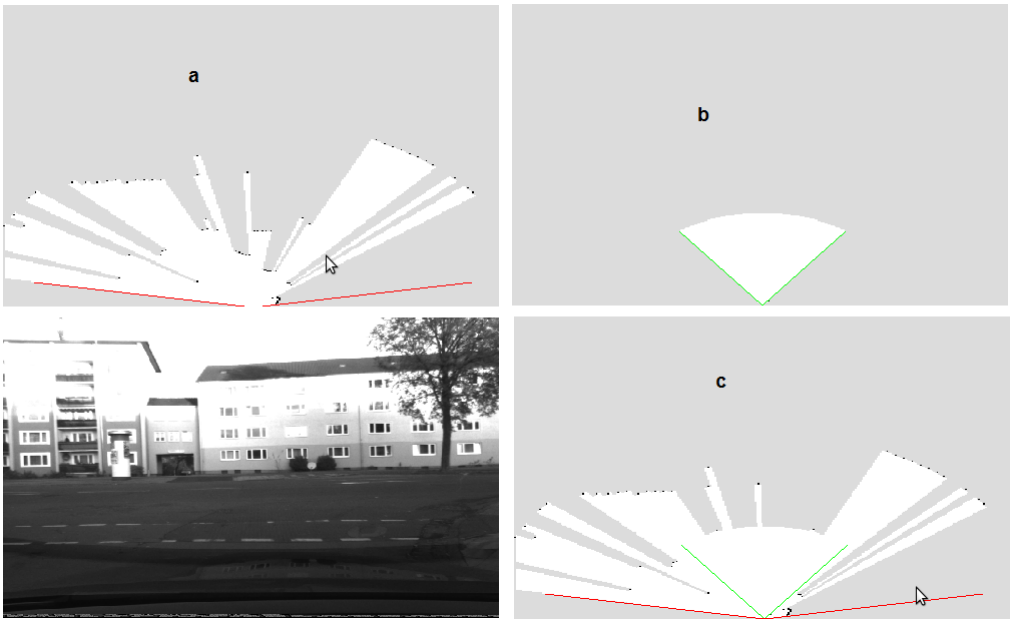

Figure 3.8: Fusion results for a scenario when false laser reflections are observed due to white markings on the empty road making them appear as objects (a). However vision sees no object (b), the fused view (c) removes the false objects observed by laser.

Although low level fusion is helpful in some cases as shown in the results to remove some problems of one sensor with the help of other one. However, usually, low level

Figure 3.9: Fusion results for a real road scenario.

fusion is too early and has two main implications: first, for all the sensors involved we need to find a way to represent data in a compatible form even if the sensors have different characteristics. Second, no sensor specific knowledge can be used after that data has been represented in this common form, for example when occupancy grid has been created from vision 3D points no vision cues can be used to segment objects from this representation.

3.2.5 Input to SLAM

As shown in figure 3.2, the output of the low level fusion is used as input for the *SLAM* module, so the fused grid servers as input in this case. More precisely the fused grid OG_f becomes z_t for equations 3.1.15 and 3.1.16. Since this fused grid has the same size and position as that of *SLAM* map so there is a one to one correspondence between cells of map and fused grid making scan matching and map update even easier.

In actual implementation, using fused grid and map built so far, we calculate the number of times a cell is observed free and number of times it is observed as occupied. If we call these numbers as N_{free} and N_{occ} respectively then the updated cell occupancy probability derived from equation 3.1.14 is given as:

$$P(m|x_{1:t}, z_{1:t}) = \frac{\exp\left(N_{occ}p_{occ} + N_{free}p_{free}\right)}{1 + \exp\left(N_{occ}p_{occ} + N_{free}p_{free}\right)} \tag{3.2.7}$$

Similarly for localization current fused occupancy grid is matched with the map from

different sampled x_t positions. The position giving best match is taken as true vehicle position.

3.3 Object level fusion

In this section we explain our object detection level fusion strategy between laser and stereo vision, this work is published as (Baig *et al.*, 2011). The position of this fusion level in the perception architecture is shown in figure 3.10. Our objective here is to

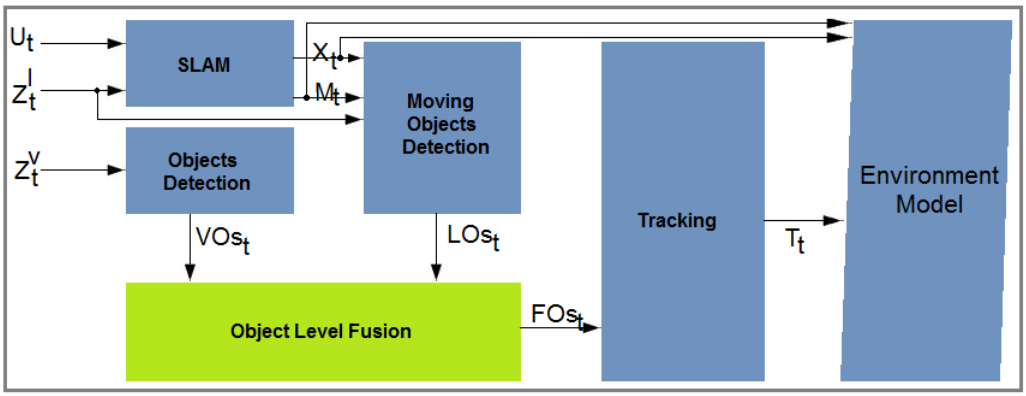

Figure 3.10: Object level fusion with perception architecture. Here VOs_t is the list of objects detected by vision, LOs_t is the list of moving objects detected by laser and FOs_t the list of fused objects.

explain following fusion function introduced in last chapter:

$$FOs_t = \mathrm{F}_{\mathrm{OL}}(VOs_t, LOs_t) \tag{3.3.1}$$

In the following we start with explaining how to get list of moving objects from laser and list of objects from stereo vision and then how we fuse these two lists to get a list of fused objects.

3.3.1 Laser Processing - Moving objects extraction

This process of extracting moving objects from laser data consists of following steps: first we construct a local grid map and localize the vehicle in it, then using this map we classify individual laser beams in the latest data frame as belonging to moving or static parts of the environment, finally we segment this laser scan to make objects from individual laser hit points.

Environment Mapping & Localization

As explained in the beginning of this chapter we have used an incremental mapping approach based on laser scan matching algorithm to build a local vehicle map. Based on occupancy grid representation the environment is divided into two dimensional lattice of rectangular cells and we keep track of probabilistic occupancy state of each cell. We build a grid map of $90m \times 108m$ with each cell having dimensions of $0.3m \times 0.3m$. Environment mapping is essentially the estimate of posterior probability of occupancy $P(m \mid x_{1:t}, z_{1:t})$ for each cell of grid m, given observations $z_{1:t} = \{z_1, ..., z_t\}$ from time 1 to time t at corresponding known poses $x_{1:t} = \{x_1, ..., x_t\}$, here $z_t = \{P_i\}$ where $P_i = (x = r_i \cos\theta_i, y = r_i \sin\theta_i)$ is the impact point of ith laser beam (from its polar coordinates r_i, θ_i) for $\forall i \leq 161$ expressed in laser frame of reference, and $x_t = (x, y, \theta)$ is the vehicle pose. To solve localization problem we have used importance sampling based particle filter Thrun *et al.* (2005). A total of 300 particles are used. For the given previous pose x_{t-1} and current odometry information $u_t = (\nu, \omega)$ (translational and rotational velocities) we sample different possible positions of the vehicle from the motion model $P(x_t|u_t, x_{t-1})$. Then we compute the probability of each position using laser data and a sensor model. The pose of the particle getting highest probability is taken as true pose.

Moving & Static Parts Distinction

By projecting the latest laser data onto the local map constructed so far, the impact points are classified as static or dynamic. The points observed in free space are classified as dynamic whereas the rest are classified as static. More precisely, if we represent the local grid map at time t as $M^t = \{m\}$ where m is a grid cell and if $M^t[P_i]$ gives the occupancy probability of the grid cell corresponding to the laser impact point P_i then we classify the laser impact points into following two types:

- $MovingPoints = \{P_i \mid M^t[P_i] < 0.5\}$
- $StaticPoints = \{P_i \mid M^t[P_i] \geq 0.5\}$

Laser Objects Extraction

Finally we perform segmentation to extract objects from these laser impact points. We define an object as:

$$\acute{O}_L = \{P_n \mid dist(P_n, P_{n-1}) < S_{thr}\}$$

Here P_i is the impact point as defined above , $dist(P_n, P_{n-1})$ is the euclidean distance between two adjacent points, and S_{thr} is the segment threshold distance which is equal to 2.0 meters in our experiments. An object is marked as dynamic if $\acute{O}_L \cap MovingPoints \neq \varnothing$, Finally we calculate the polar coordinates of center of gravity (centroid) (r_L, θ_L) of each object using Cartesian coordinates of its constituting points as: $r_L = \sqrt{\acute{x}^2 + \acute{y}^2}$ and $\theta_L = atan2(\acute{y}, \acute{x})$ where $\acute{x} = \sum_i x/n$ and $\acute{y} = \sum_i y/n$ for $\forall P_i(x, y) \in \acute{O}_L$ and $n = |\acute{O}_L|$.

Laser Processing Output

The output of laser processing step at time t consists of a local grid map M^t, and a list of detected moving objects $L^t_{objects} = \{O_L\}$ where $O_L = (\acute{O}_L, r_L, \theta_L)$ is moving object with centroid information. Grid map is only used to display on the screen whereas list of dynamic objects is used further for fusion. The results of laser processing are shown in Figure 3.11.

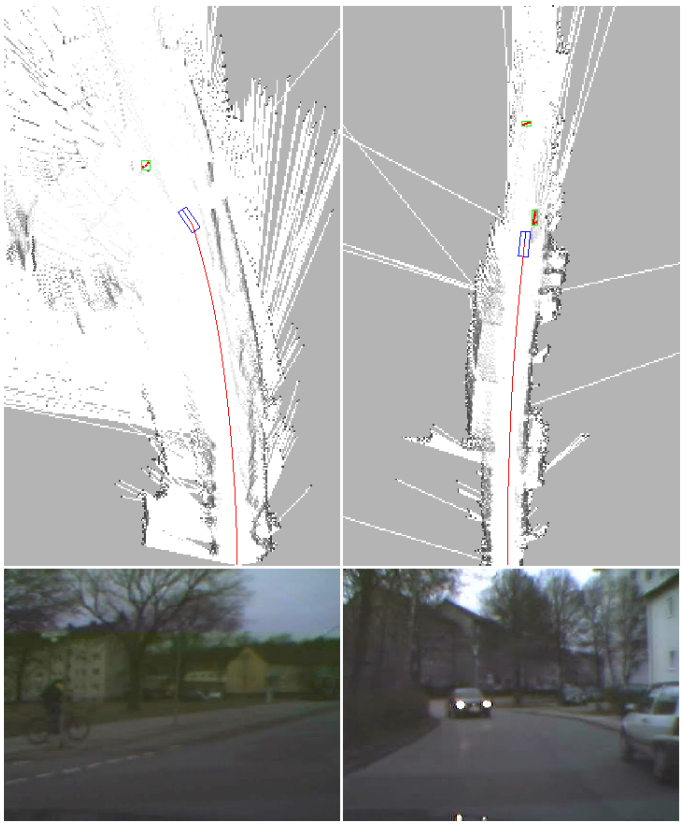

Figure 3.11: Mapping and moving objects detection results. Detected moving objects (a bicycle in left image and two cars in right image) are shown as green rectangles.

3.3.2 Stereo vision processing

As already mentioned in 1st chapter, another team from University of Cluj-Napoca Romania working on INTERSAFE-2 project has performed the stereo vision processing Nedevschi *et al.* (2009a,b) (their processing details are summarized in chapter 5). Their output of stereo vision processing consists of a list of objects detected in each frame of the stereo images. For each object in the list we are given 3D coordinates of the four corners of the lower rectangle of the object cuboid in stereo frame of reference, the height of the top rectangle and the class of the object (pedestrian, pole vehicle etc), but no information about the dynamic state of the object are available, objects can be dynamic or static. From this data we can easily calculate the coordinates of the eight corners of the cuboid.

Pre-Fusion Processing

We need to perform some processing on stereo objects before performing fusion. The first step before performing fusion between laser data and stereo vision objects is to project stereo objects onto the laser plane using transformation matrices to achieve the common spatial reference. For object level fusion between laser and stereo we need to represent vision objects by their centroids. We take this centroid as the middle point of the front line segment of object rectangle. Since laser points also belong to the front end of objects, so this centroid gives better results than the object rectangle center. We calculate polar coordinates of the centroid for each vision object (in a similar way as explained for laser objects) to make the representation compatible to the laser objects for fusion.

Stereo Vision Processing Output

The output of stereo vision processing at time t consists of a list of objects $V^t_{objects} = \{O_V\}$ where $O_V = (r_V, \theta_V, class)$. r_V and θ_V are the polar coordinates of the object centroid. The set of classes consists of $\{pedestrian, vehicle, cyclist, pole\}$.

3.3.3 Fusion

In this section we give details of object detection level fusion between laser and stereo vision sensors. Two lists of objects are input to the fusion process: list of dynamic objects detected by laser and represented as centroid points, and list of objects (static or dynamic) detected by stereo vision represented as points along-with classification

information. We believe that an object detection level fusion between these two lists can complement each other thus giving more complete information about the states of objects in the environment. This fusion process consists of following two steps:

Object Association

In this step we determine which stereo objects are to be associated to which laser objects from the two object lists, using nearest neighbor technique. The positional uncertainty of an object given by stereo vision increases with depth, so we have defined a distance threshold function based on the depth of the stereo object from the origin as:

$$V_{thr} = 5 * \frac{r_V}{36}$$

V_{thr} is the uncertainty in position of an object detected at a distance of r_V by stereo vision. Here 5 (meters) is the maximum depth uncertainty for an object detected at a distance of 36 meters. Stereo objects beyond this distance are ignored because the effective range of stereo is limited to $36m$ for this work. A stereo object O_V^i is associated to a laser object O_L^j if $dist(O_V^i, O_L^j) < V_{thr}$ and O_L^j is closest to O_V^i.

Position information fusion

This step works on the pair of objects associated with each other in the previous step and fuses their position (range and bearing) information. We model the position uncertainty using 2D Gaussian distribution for both objects. Suppose $P_L = [r_L, \theta_L]^T$ is the centroid position of laser object and $P_V = [r_V, \theta_V]^T$ is the centroid position of associated stereo vision object. If X is the true position of the object then the probability that laser detects this object at point P_L is given as:

$$P(P_L|X) = \frac{e^{\frac{-(P_L-X)^T R_L^{-1}(P_L-X)}{2}}}{2\pi\sqrt{|R_L|}}$$

and similar probability for stereo object is given as:

$$P(P_V|X) = \frac{e^{\frac{-(P_V-X)^T R_V^{-1}(P_V-X)}{2}}}{2\pi\sqrt{|R_V|}}$$

Here R_L is the $2X2$ covariance matrix of range and bearing uncertainty calculated from the uncertainty values provided by the vendor. Whereas R_V is the covariance matrix for stereo vision and depends on the depth of the object from origin. To calculate this matrix empirically for different ranges (with a difference of 2 meters) we manually associate vision objects with corresponding laser objects. Considering the laser object

position as the mean true position of the object we are able to calculate their difference for both range and bearing values for vision objects. Repeating this process for different ranges in different data sets we are able to collect data for calculating this matrix.

Using Bayesian fusion the probability of fused position $P = [r_F, \theta_F]^T$ is given as:

$$P(P|X) = \frac{e^{\frac{-(P-X)^T R^{-1}(P-X)}{2}}}{2\pi\sqrt{|R|}}$$

where P and R are given as:

$$P = \frac{P_L/R_L + P_V/R_V}{1/R_L + 1/R_V}$$

and

$$1/R = 1/R_L + 1/R_V$$

respectively. This process of fusion is shown in figure 3.12. P is taken as the position of the fused object.

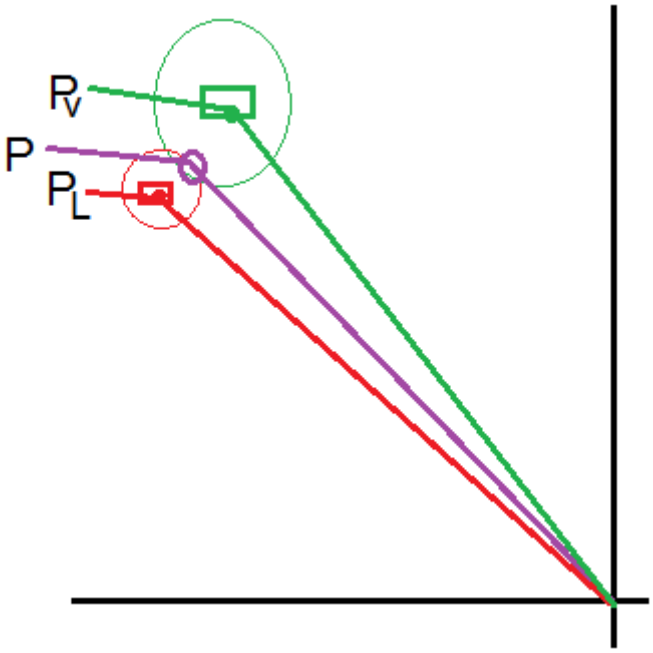

Figure 3.12: Fusion process: red color shows the position uncertainty of laser object, green color for corresponding stereo object and violet the fusion result of the two.

The result of this fusion process is a new list of fused objects. This list also has all the laser objects which could not be associated with stereo objects and all the stereo objects which could not be associated with some laser objects. We keep unassociated stereo objects because they may correspond to dynamic objects which may not have been detected by laser in current frame due to occlusion as explained next.

Figure 3.13 shows an interesting fusion scenario, objects shown in cyan color are the objects detected by stereo vision whereas the objects shown by light violet rectangles are the laser detected objects, red dots are raw laser impact points. An oncoming car that was being detected by laser until last scan is now occluded by a cyclist whereas stereo system was able to detect this car. So this miss detection by laser will be filled in by stereo during fusion hence giving a smooth track. The increased position uncertainty with depth for stereo vision objects can also be seen in the figure (green ellipses).

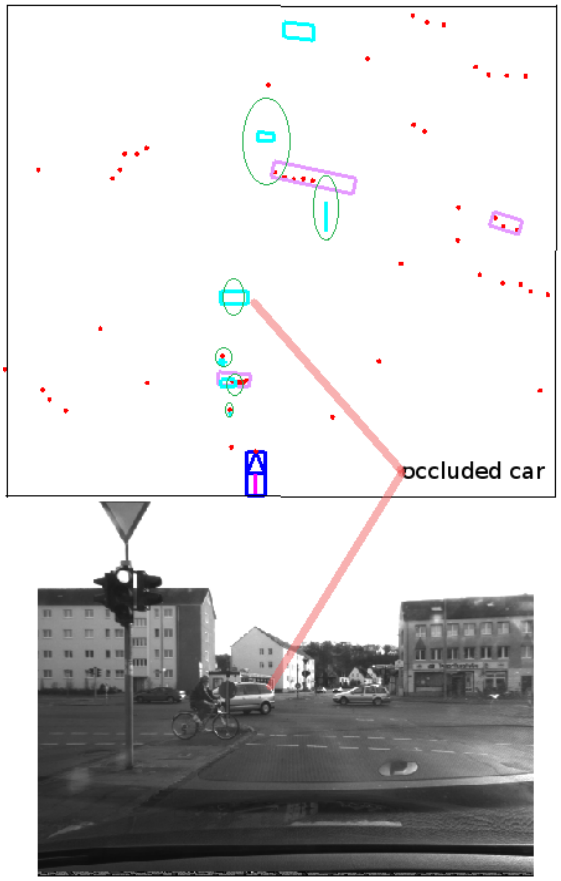

Figure 3.13: Laser and Stereo vision objects fusion, see the text for details.

Fusion Output

The output of fusion process consists of fused list of objects $F^t_{objects} = \{O_F\}$ where $O_F = (r_F, \theta_F, class, DynamicState, SensorCount)$. For each object we have position (centroid) information, dynamic state information (dynamic or unknown, unknown for unassociated stereo objects), classification information and a count for number of sensors detecting this object. Sensor count value is especially useful during tracking, if an object is detected by two sensors then we create a track immediately instead of waiting for some frames to confirm the object as real.

3.3.4 Tracking

In general multi objects tracking problem is quite complex. It includes the definition of tracking methods, association methods and maintenance of objects currently present in the environment. Usually Bayesian filters are used to solve this problem which require the definition of a specific motion model of tracked objects to predict their positions in the environment. Using the prediction and observation update combination, new position of each object is estimated. In the following we explain the components of our tracking module.

Data Association

This step consists of assigning new objects of fused list to the existing tracks. Since in the current work we are more concerned with tracking multiple objects in an intersection like scenario so it is important to choose a more effective technique of data association. We have used MHT (Reid, 1977) approach to solve the data association problem. An important optimization that we have achieved here due to fusion process mentioned above is related to classification information provided by stereo vision. While generating hypotheses we ignore all those hypotheses which involve objects from different classes. For example a hypothesis trying to involve a pedestrian with a vehicle in a track will be ignored, this significantly reduces the number of hypotheses. To further control the growth of tracks trees we need to use some pruning technique. We have chosen the N-Scans pruning technique to keep the tracks trees to a limit of N.

Track Management

In this step tracks are confirmed, deleted or created using the m-best hypotheses resulting from the data association step. New tracks are created if a new track creation hypothesis appears in the m-best hypothesis. A newly created track is confirmed if it is updated by objects detected in current frames after a variable number of algorithm steps (one step if the object was detected by both laser and stereo vision otherwise in three steps). This implies that the spurious measurements which can be detected as objects in the first step of our method are never confirmed. To deal with non-detection cases, if such a hypothesis appears (which can appear for instance when an object is occluded by another one) tracks with no new associations are updated according to their last position, for them next filtering stage becomes a simple prediction. In this way a track is deleted if it is not updated by a detected object for a given number of steps.

Filtering

Since in an intersection like scenario there may be different types of objects (vehicles, motor bikes, pedestrians etc) moving in different directions using different motion modes, a single motion model based filtering technique is not sufficient. To address the tracking problem in this scenario we have used an on-line adapting version of Interacting Multiple Models (IMM) filtering technique. The details of this technique can be found in our other published work Vu *et al.* (2009). We have seen that four motion models (constant velocity, constant acceleration, left turn and right turn) are sufficient to successfully track objects on an intersection. We use four Kalman filters to handle these motion models. Finally the most probable trajectories are computed by taking the most probable branch and we select one unique hypothesis for one track tree.

3.3.5 Results

Fusion and tracking results are shown in figures 3.14 and 3.15 along-with the images of corresponding scenarios. Figure 3.14 shows an interesting intersection scenario for

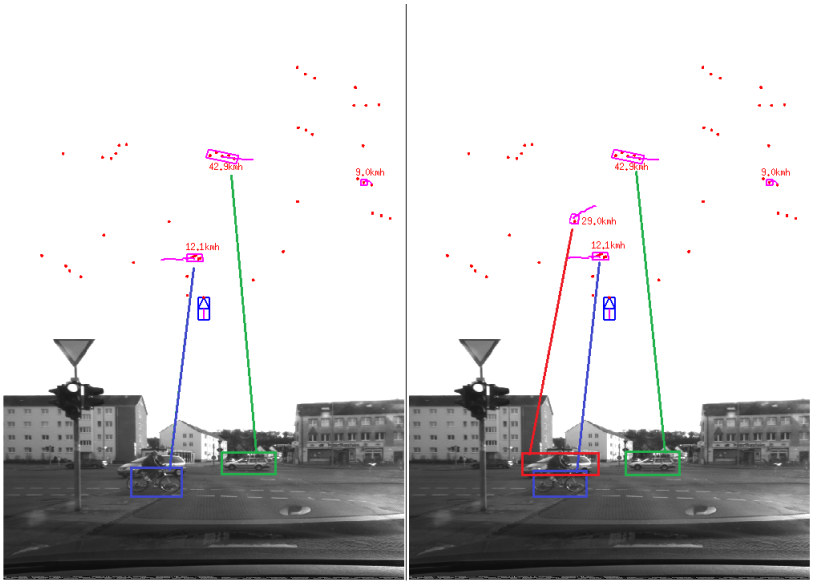

Figure 3.14: Fusion and tracking results. Left(laser only): car occluded by cyclist is not being tracked. Right(laser and stereo): with fusion car was successfully tracked.

fusion. Here left image shows tracking results based only on laser data, car behind the cyclist was occluded in last few frames giving insufficient impact points to be detected as a moving object. Right image shows tracking with fusion, car was also partially detected by stereo vision and was successfully tracked in the fused results. Figure 3.15 shows a similar scenario, the ego vehicle is waiting for the signal, a truck turning left,

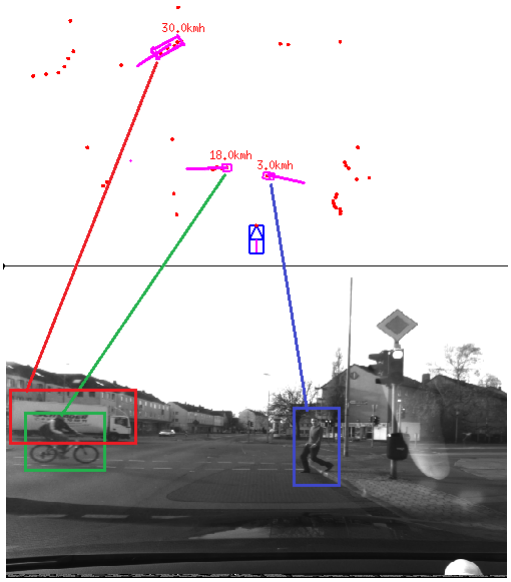

Figure 3.15: Tracking results for a truck, a pedestrian and a cyclist.

Table 3.1: Data sets used for experiments for this work.

	Data Set 1	Data Set 2	Data Set 3
Scenario	Urban road and intersection driving	Urban road and intersection driving	Urban road and intersection driving
Time	1 Min 4 Sec	2 Min 30 Sec	3 Min 16 Sec
True Moving Objects	30	21	43

a cyclist and a pedestrian crossing the road in opposite directions are being tracked. Although truck in this scenario is partially occluded by the cyclist but due to fusion it has been tracked successfully. Table 3.2 shows empirically observed statistics of missed tracks for three data sets given in table 3.1.

We have observed that in many of the cases object level fusion is the more appropriate level of fusion. It is not too early that sensor specific information are ignored and not too late that even real objects are ignored due to lake of sustained evidence over several data frames.

Table 3.2: Statistics of missed tracks with and without fusion for three data sets.

Data Set	Missed tracks without fusion	Missed tracks with fusion
1	10	7
2	12	8
3	7	4

3.4 Track level fusion

Position of this fusion level in the perception architecture is shown in figure 3.16. For this level of fusion moving objects are detected and tracked for each sensor separately and fusion is performed on the lists of tracked objects (called tracks). The objective of fusion at this level is to be more certain about the presence of moving objects near the host vehicle moving at fast speed. In the following we first summarize the sensor processing for both sensors to get tracks and then give the fusion scheme used to fuse lists of these tracks given by individual sensors (laser and stereo vision).

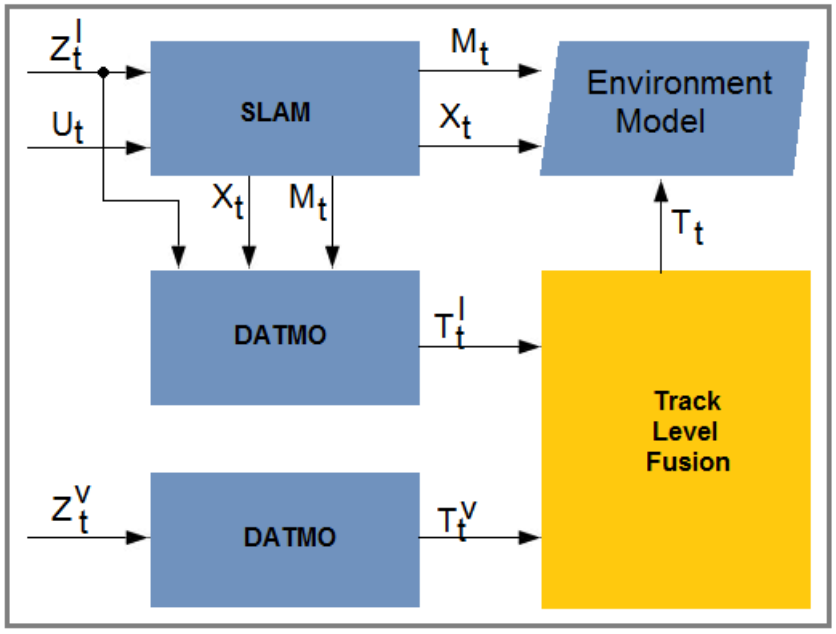

Figure 3.16: Track level fusion with perception architecture. Here T_t^l is the list of tracks detected by laser, T_t^v is the list of tracks detected by stereo vision and T_t the list of fused tracks.

3.4.1 Sensor processing

Lidar processing

Lidar processing done for this level is the same as explained in previous two sections. We first build a grid based map of the environment and localize the host vehicle in this map using maximum likelihood *SLAM* solution. Then using the map we find the moving laser points by projecting new frame on the map and finding the inconsistencies of these projected points with the built map. In next step we segment the detected moving points to form objects. Finally we track these objects using the MHT and IMM based techniques as explained in previous section. This lidar processing gives us a set of tracks T^l. Figure 3.17 shows track results from lidar data for two cars.

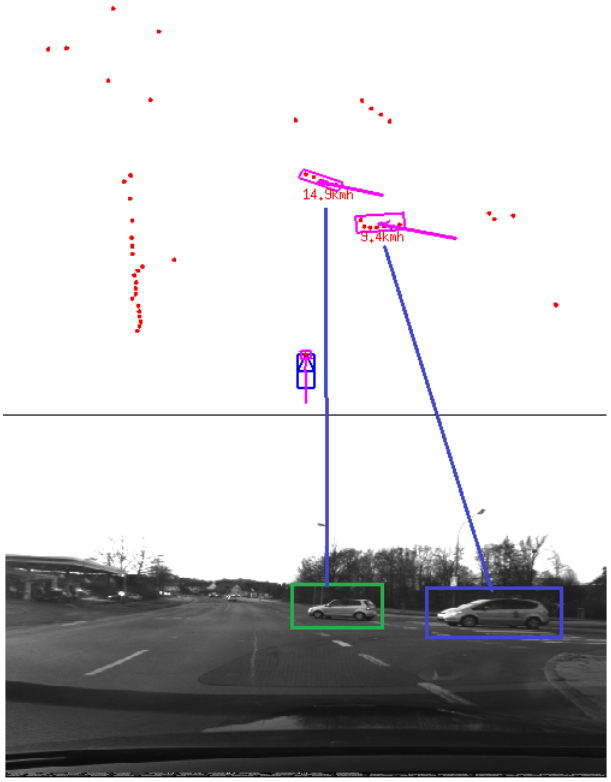

Figure 3.17: Tracking results for two cars.

Stereo processing

As mentioned in previous section, output of the stereo vision processing by Cluj team consists of a list of 3D objects for each data frame. However, this list contains both static and dynamic objects and no information is provided if an object is static or dynamic. To solve this problem we consider all objects as dynamic and try to track them using

a similar MHT and IMM based scheme as that of laser (using localization information given by laser processing). Due the the absence of dynamic state information of objects usually many false positives are reported. Some tracking results based on this scheme for stereo vision are shown in figure 3.18.

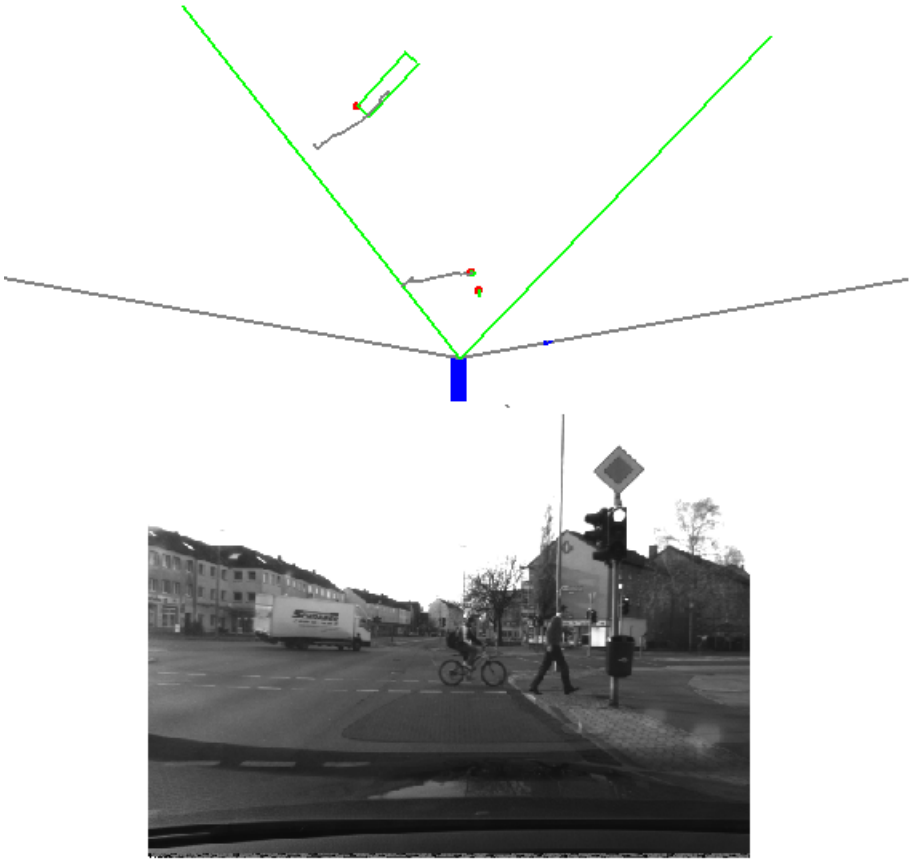

Figure 3.18: Stereo vision tracking results of a truck, cyclist and pedestrian.

3.4.2 Fusion scheme and results

Fusion at this level mainly involves associating tracks from both individual sensors and computing their compatibility. We have used a gating approach based on the covariance matrices of the tracks for both sensors, similar to Bar-Shalom and Chen (2004) using statistical distance. If two tracks pass this association check then we perform an additional step to check their trajectory consistency, the main difference of our approach with others. Associated tracks having parallel trajectories are confirmed as true tracks and are fused together. We check this parallelisms as follows:

Suppose that two tracks T^l and T^v fall in same gate, each of these tracks consists of some of its previous poses (position and orientations) and some other information (like

velocity, acceleration). If T^l has N_l previous poses and T^v has N_v previous poses then we can represent these tracks as (by ignoring velocity and acceleration information etc for simplicity sake):

$$T^l = \{(x^l_t, y^l_t, \theta^l_t), (x^l_{t-1}, y^l_{t-1}, \theta^l_{t-1}), \ldots, (x^l_{t-N_l}, y^l_{t-N_l}, \theta^l_{t-N_l})\}$$

$$T^v = \{(x^v_t, y^v_t, \theta^v_t), (x^v_{t-1}, y^v_{t-1}, \theta^v_{t-1}), \ldots, (x^v_{t-N_v}, y^v_{t-N_v}, \theta^v_{t-N_v})\}$$

If $N = min(N_l, N_v)$ then:

$$\triangle\ \theta = \frac{\sum_{i=0}^{N} |\theta^l_{t-i} - \theta^v_{t-i}|}{N}$$

$$\triangle\ x = \frac{\sum_{i=0}^{N} |x^l_{t-i} - x^v_{t-i}|}{N}$$

$$\triangle\ y = \frac{\sum_{i=0}^{N} |y^l_{t-i} - y^v_{t-i}|}{N}$$

Two tracks T^l and T^v are considered parallel if $\triangle\ \theta < \theta_{thr}$, $\triangle\ x < x_{thr}$ and $\triangle\ y < y_{thr}$ where these threshold values are determined empirically. Figure 3.19 shows this concept of parallel tracks.

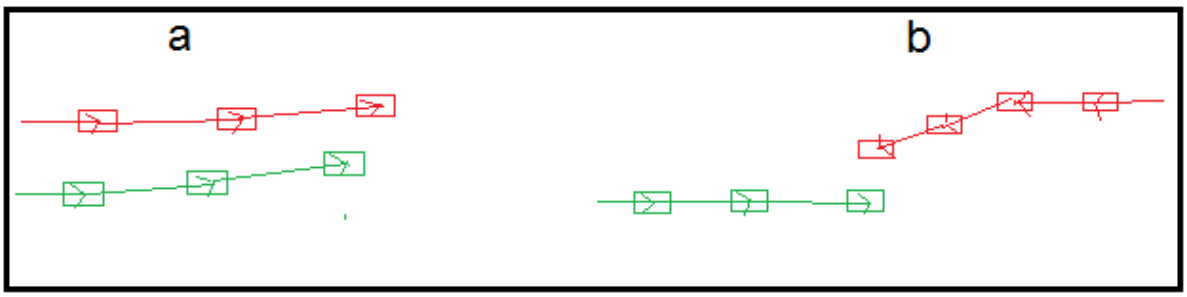

Figure 3.19: Illustration showing the concept of parallel tracks. (a) shows a track detected in laser data (red) and a track detected in stereo vision data (green). These tracks are considered as parallel as they fulfill the $\triangle$ constraints, and will be fused. However, two tracks shown in figure (b) are not parallel as they are moving in opposite directions and will not be fused.

The fused list of tracks has two types of tracks: fused tracks, tracks visible in fields of view of both laser and stereo vision and passing the parallel check, and the tracks detected by laser beyond the stereo vision range. Tracks of first type carry more weight.

Another layer of processing can be implemented on this fused list of tracks using some data association and filtering technique to predict new position of fused tracks and then to update the properties of tracks using observations consisting of tracks from individual sensors. In this case parallelism check can be used to reduce number of hypothesis during data association step.

Table 3.3: Statistics of tracks detected for laser and stereo vision data and tracks passing the parallel test.

Data Set	Laser Tracks	Stereo Tracks	Parallel Tracks
1	108	24	18
2	147	38	27
3	198	65	54

This fusion strategy helps remove false tracks detected by laser within the stereo vision view. Some tracking results are shown in figure 3.15 and 3.17. Some empirically observed results are shown in table 3.3 for data sets given in table 3.1. We observed that the tracks passing the parallel test were 98% true tracks.

Different track output schemes can be used depending on the sensor noise levels. For too much noisy sensors only tracks falling in fields of view of both sensors and fulfilling the parallel check are output, however this scheme usually has many miss detections. Another scheme can be to output a union of both track lists with a tag showing whether a track was detected by a single sensor or by both (and passed the parallel check).

One advantage of fusion at this level is that it can be implemented as independent module without requiring knowledge of sensor specific information. This module only needs lists of tracks as input to give fused list of tracks.

3.5 Road border detection

In this section we present a fast algorithm for detection of road borders from noisy laser data. We have observed that the laser scanner on demonstrator used for current work has following problems due to noise: Data appear vibrating and have reflections from white markings on the road. On the right side near the right edge of the host vehicle especially, the edges of road border appear moving and tend to be curved. It also has longitudinal uncertainty that causes the objects to appear as clutter of points rather than having some uniform outline. From these observations we learned that a successful detection of road borders in laser data can help remove a significant number of false alarms detected on and beyond road borders. This road border detection should be efficient enough to let both SLAM and DATMO run on-line in real time without requiring any extra hardware modifications. In this work we present a simple to implement but efficient road border detection algorithm and use it to reduce false alarms in our DATMO solution.

In this section we present details of our road border detection technique which consists of representing the grid map in an intermediate form called hit grid, removing noise, defining a search window, a score function and details about how these search windows are evaluated in an efficient way to find road border.

Since we do not use special setup of laser scanner for detecting road borders, our technique is based on the hypothesis that road curbs are frequently detected by the laser mounted on the host vehicle while driving on the road. If there are parked cars alongside the road we can effectively take them as road border markers or end of drivable road area. Our road border finding process consists of following steps:

1) *Hit grid (HG) representation*; along with the occupancy grid of local map we generate another grid, called hit grid. This grid has same number of cells and spans the same area as that of occupancy grid but each cell of this grid keeps only a count of laser hits detected in area corresponding to this cell. With each new scan reading from laser scanner the count of hits keeps on increasing in cells corresponding to static parts of environment. An advantage of this grid is that, no complicated update formulas need be evaluated for each new observation. The update step only involves incrementing the count of a cell where a laser hit is detected in new observation. If we represent this HG at time instance t as H^t then the value of a cell at the intersection of row i and column j is represented as H^t_{ij}. The update process for this grid consists of finding corresponding cell H^t_{ij} for each laser hit point $z^t_{k=1:161}$ and incrementing its count.

2) *Noise removal*; to filter false hits caused by sensor error and moving objects we perform two operations on each cell of the grid: first, we do the smoothing by adding to the value of each cell the counts of its eight neighbors (a classic smoothing technique in vision), second, we pass this new grid, denoted as S^t and called smoothed hit grid, through a high pass filter. We reset the count value to zero for those cells of S^t which have values less than filter's cut-off value defined as $F_{thr} = Max(S_{ij})/10$. This removes the erroneous values and the cells having count values greater than zero belong to areas having static objects including road borders. Formally these operations are given as:

$$S^t_{ij} = \sum_{i-1}^{i+1} \sum_{j-1}^{j+1} H^t_{kl} \tag{3.5.1}$$

$$S_{ij} = \begin{cases} S_{ij} & \text{if } S_{ij} >= F_{thr} \\ 0 & \text{otherwise} \end{cases} \tag{3.5.2}$$

3) *Search window*; since we suppose that host vehicle is always (excluding when it is on an intersection) oriented parallel to road border, so instead of searching road border everywhere in the grid we search only specific areas in the hit grid. These areas to be

searched lie on left and right side of the host vehicle. So we define two search windows of length l and width w (in our experiments $l = 20$ and $w = 1$, in meters, give good results), one on left and the other on right side of the host vehicle. Initial positions of these windows are at 1 meter on right and 1 meter on left of host vehicle. Then in each search iteration the left window is moved 0.9 meter left (maximum 25 iterations) and right window is moved 0.9 meter right (maximum 15 iterations) and a score function is evaluated for the hit counts present in the cells lying under search window. Finally a threshold value is used to decide the presence or absence of road border. In each direction the first window that shows the presence of a road border stops the search in that direction. Figure 3.20 shows host vehicle along with some search windows in the smoothed hit grid.

4) *Score function*; we have defined a very simple score function that can be evaluated very fast. According to this function the score of a search window W consists of the sum of hits of all cells lying under this window:

$$Score(W) = \sum S_{ij}^t | S_{ij}^t \in W \tag{3.5.3}$$

5) *Detailed algorithm*; Before going into our model details we will define following mathematical notations (as defined in Smith and Cheeseman (1987)): The pose (position with orientation) of an object j (real or hypothetical) with respect to a reference frame i is represented as $P_{ij} = [x_{ij}, y_{ij}, \theta_{ij}]^T$. We also define following compounding pose relations for pose transformations: if P_{ij} is the pose of object j w.r.t object i and $P_{jk} = [x_{jk}, y_{jk}, \theta_{jk}]^T$ is the pose of object k w.r.t j then the pose of k w.r.t i denoted as $P_{ik} = [x_{ik}, y_{ik}, \theta_{ik}]^T$ is given as:

$$P_{ik} \equiv \oplus(P_{ij}, P_{jk}) = \begin{bmatrix} x_{jk}\cos(\theta_{ij}) - y_{jk}\sin(\theta_{ij}) + x_{ij} \\ x_{jk}\sin(\theta_{ij}) + y_{jk}\cos(\theta_{ij}) + y_{ij} \\ \theta_{ij} + \theta_{jk} \end{bmatrix}$$

For the pose P_{ij} the reverse pose relationship $P_{ji} = [x_{ji}, y_{ji}, \theta_{ji}]^T$ (pose of i w.r.t j) is defined as:

$$P_{ji} \equiv \ominus(P_{ij}) = \begin{bmatrix} -x_{ij}\cos(\theta_{ij}) - y_{ij}\sin(\theta_{ij}) \\ x_{ij}\sin(\theta_{ij}) - y_{ij}\cos(\theta_{ij}) \\ -\theta_{ij} \end{bmatrix}$$

Considering figure 3.21, our objective is to find cells lying under red rectangle (our search window) so that score function can be evaluated for them. To proceed we have three frames of reference: grid frame of reference with origin O_g shown in color green, laser frame of reference with origin O_l shown in color blue and search window frame of reference with origin O_r shown in color dark red. The pose of O_l w.r.t O_g is known

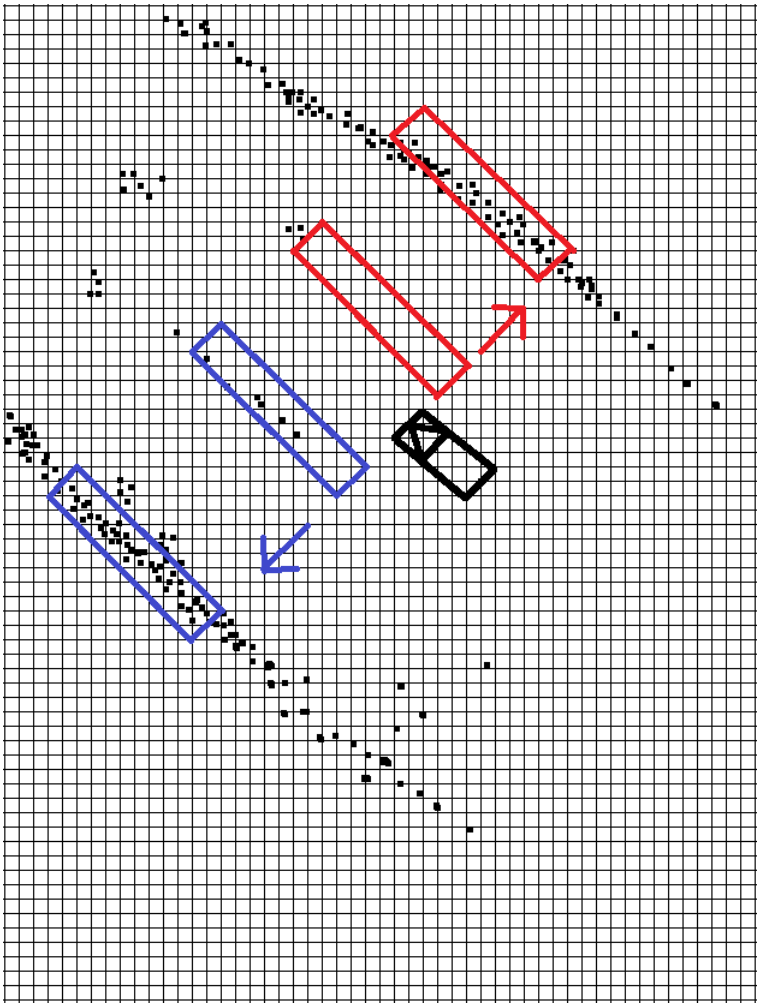

Figure 3.20: Position of host vehicle and two left (blue) and two right (red) search windows in smoothed hit grid. The two search windows for each left and right side include the initial window and the one with highest score. Windows with highest score lie on road border. Cells having black dots have hit count values non zero. Arrows show the direction of movement of their respective windows.

and is $P_{gl} = [x_{gl}, y_{gl}, \theta_{gl}]^t$. Similarly pose of O_r w.r.t O_l denoted as $P_{lr} = [d, 0, 0]^t$ is also known, because we always take it on X_l at a distance d (that increases with each search iteration by 0.9 meters).

Coordinates of four corners of red rectangle w.r.t O_r are fixed and given as $(-0.5, 0)$, $(0.5, 0)$, $(-0.5, 20)$ and $(0.5, 20)$. So a point $P(x_{lp}, y_{lp})$ lies in red rectangle if $-0.5 <= x_{lp} <= 0.5$ and $0 <= y_{lp} <= 20$. Each cell can be represented by its middle point $P = [x_{gp}, y_{gp}]^t$, the coordinates of this point can be easily calculated w.r.t O_g from its row and column using following equations

$$x_{gp} = col * CellSize + CellSize/2 \tag{3.5.4}$$

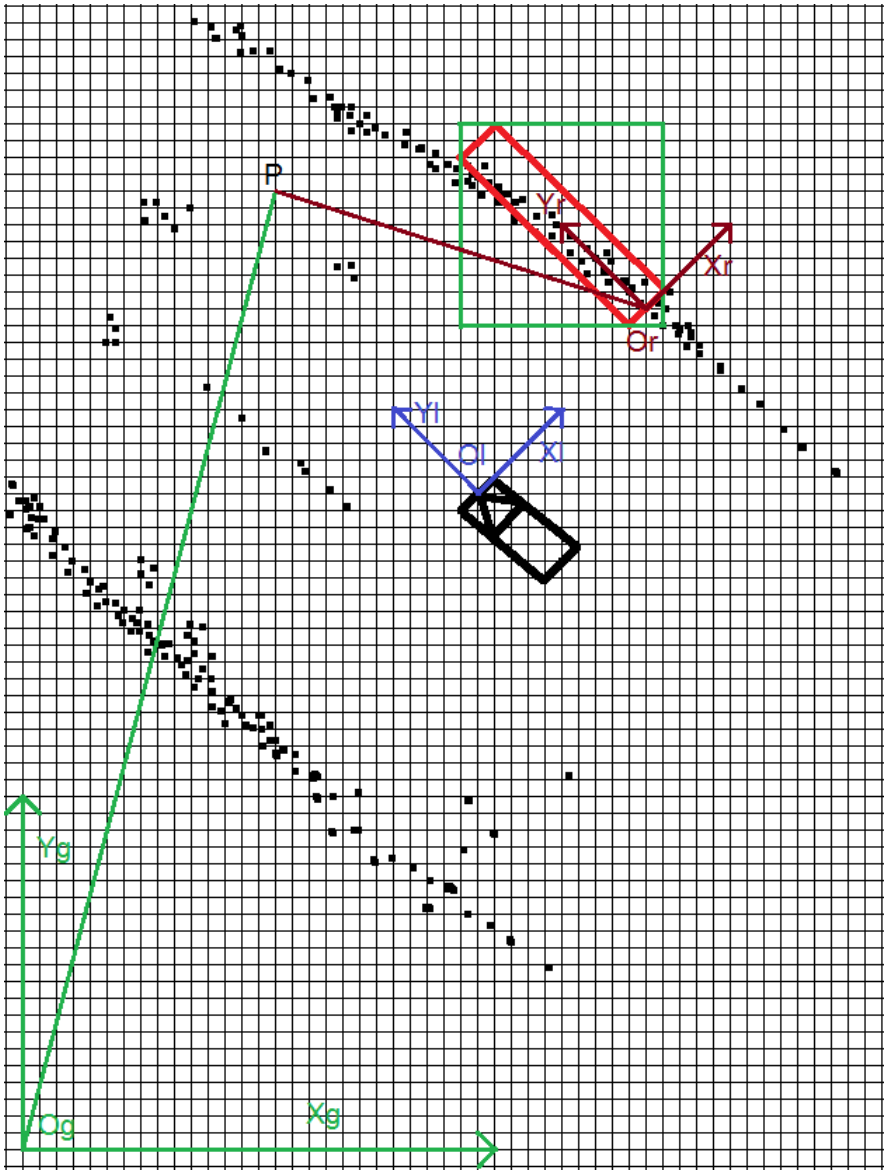

Figure 3.21: Evaluation of score function for red rectangle. Three reference frames belong to smoothed hit grid O_g, laser frame of reference O_l, and search rectangle frame of reference O_r. Our objectives are to find the coordinates of point P, given in O_g reference frame, w.r.t O_r reference frame and to find row and column of bottom left and top right corners of green rectangle. Cells within green rectangle will be searched only to find the cells lying under red rectangle.

$$y_{gp} = row * CellSize + CellSize/2 \tag{3.5.5}$$

But to calculate the coordinates of same point P w.r.t O_r i.e $P = [x_{rp}, y_{rp}]^t$ we need following operations: Pose of O_r w.r.t O_g:

$$P_{gr} = \oplus(P_{gl}, P_{lr}) \tag{3.5.6}$$

Pose of O_g w.r.t O_r:

$$P_{rg} = \ominus(P_{gr}) \tag{3.5.7}$$

Pose of P w.r.t O_r:

$$P_{rp} = \oplus(P_{rg}, P_{gp}) \tag{3.5.8}$$

Using these relations we can calculate the coordinates of middle point of any grid cell w.r.t O_r and then we can easily check if this point lies in the red rectangle or not.

Now we have the tool to know if a cell lies under red rectangle or not. So to evaluate the score function one brute force method can be to loop on all cells of the grid, evaluate above equations on them and sum the score for cells which are in red rectangle and ignore the others. But it will be quite inefficient to evaluate all the cells when the red rectangle is very small as compared to whole grid. To solve this problem we will restrict our search to cells in green rectangle, but first we need to calculate row and column of bottom left and top right corners of this rectangle. To calculate this, suppose $P_{rp} = [x_{rp}, y_{rp}, 0]^t$ is the pose of a corner of red rectangle w.r.t O_r the pose of same corner w.r.t O_g can be calculated using following relation:

$$P_{gp} = \oplus(P_{gr}, P_{rp}) \tag{3.5.9}$$

This way we can calculate coordinates of all four corners of the red rectangle w.r.t O_g. Then minimum of the four x and y values will give bottom left corner of green rectangle w.r.t O_g and maximum of the four x and y values will give top right corner. Finally using reverse transforms of equations 3.5.4 and 3.5.5 we can calculate rows and columns of cells on these corners.

The window giving highest score greater than a threshold value is taken as belonging to road border and the line connecting the middle points of this search rectangle (red) along the length is taken as road border.

3.5.1 Applying road borders to detect false positives

After we have detected the road borders, the process of removing false positives from the detected moving objects in the current laser scan consists of following steps:

1) See if the detected object lies to the left or to the right of the host vehicle. This is important to check it against appropriate road border.

2) Transform the rectangle (only four corner points) of the detected moving object to the reference frame of the search window of the detected road border on the left or right side depending on the inferred position of the object in step 1. To do this, suppose the

pose of one corner point P of the object rectangle w.r.t O_g is $P_{gp} = [x_{gp}, y_{gp}, 0]^t$. Then the pose of this point w.r.t O_r denoted as $P_{rp} = [x_{rp}, y_{rp}, 0]^t$ is given by equation 3.5.8. Then this point is a false positive if $x_{rp} >= -0.5$ for right road border or $x_{rp} <= 0.5$ for left road border. We repeat this process for all four points of object rectangle, this object is taken as valid moving object if none of its corner points is a false positive.

3) If object is a false positive then remove it from the list of detected dynamic objects before passing to tracking module.

3.5.2 Results

Figure 3.22 shows the results of smoothing process of a scene. In the smoothed hit window road borders are easily visible. Two moving cars in front of host vehicle have no traces in the smoothed grid.

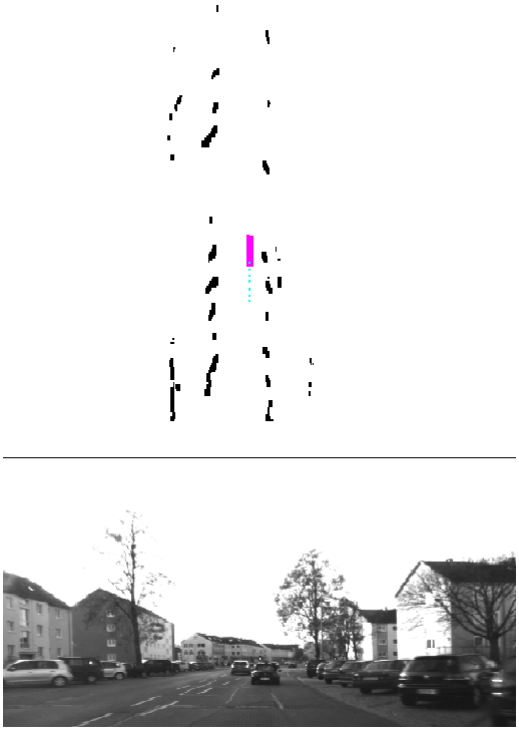

Figure 3.22: Smoothed hit grid (top) with host vehicle (pink rectangle) for an actual road scene (bottom), smoothing process removed the laser hits detected on moving cars.

Results of road border detection for two scenarios are shown in figure 3.23. The detected road borders are shown in map view instead of smooth hit grid.

Figure 3.24 shows the detected road borders in moving objects tracking view. Red dots are the current laser hits and small pink rectangles with speed captions are the detected

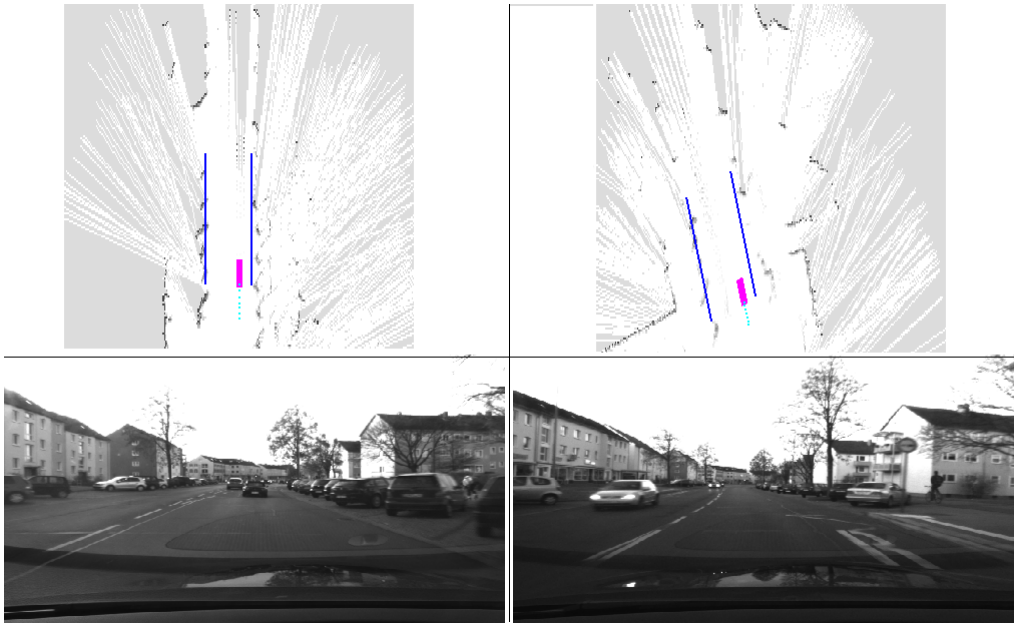

Figure 3.23: Detected road borders shown in map view for two scenarios.

moving objects by our DATMO. In both of these examples they are false alarms due to noisy laser data. It can be seen that these false positives lie on the detected road borders, which allows us to infer them as false positives and filter them out. In these figures they are shown before this filtration just to show them as false positives.

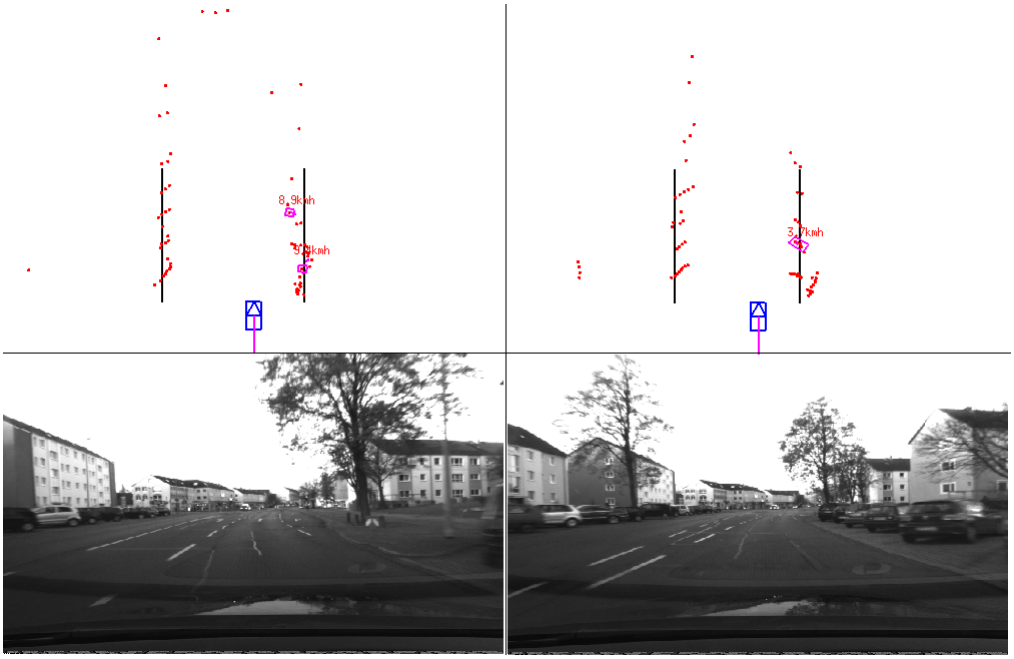

Figure 3.24: False positives (static objects detected as moving due to sensor noise) detected on road borders, with this knowledge of road borders we can successfully identify them as false positives.

Table 3.4 shows empirically observed quantitative results for detected false positives

Table 3.4: Statistics of false positives (FP) with and without road border information for three data sets.

Data Set	FP without road border	FP with road border
1	70	45
2	103	80
3	150	95

Table 3.5: Statistics of intersection detection against ground truth for three data sets.

Data Set	Total intersection	True positives	False positives
1	2	2	1
2	1	1	1
3	4	4	4

using road border information for data sets given in table 3.1.

Another important result is the detection of intersections on road. Our simple hypothesis in this regard is that, if no road borders are detected at some instance then host vehicle is on an intersection. Some states of intersection detection are shown in table 3.5.

3.6 Synthesis

In this chapter we have detailed our fusion based solution to *SLAM* and *DATMO* problems. After giving theoretical solution to these problems we explain fusion at three levels between laser and stereo vision. Maximum likelihood based *SLAM* solution that we have used is quite fast and gives good results. In this method new position of the vehicle in the map is calculated using sampling technique from its motion model. From each sampled position current data scan is matched with the map and the position giving best match is taken as the new updated position of the vehicle. The map is updated using this position and latest data scan. This technique gives good results and is fast enough to allow real time on-line calculations.

Occupancy grid based low level fusion between laser and stereo vision requires to build two grids one for each sensor and then fuse these grids using some appropriate fusion technique suitable for sensor properties. To build an occupancy grid we need to define an appropriate inverse sensor model, although for laser we have used a well known

inverse sensor model, however for stereo vision we have defined a detailed inverse sensor model that gives an occupancy grid compatible with laser grid. This fusion is useful in the scenario where one sensor has known problems that can be reduced using information from other sensor. Since laser scanner used for this work has problem of giving reflections from white markings so giving more weight to vision grid solved this problem.

Object level fusion detailed in this chapter fuses objects of two lists of moving objects detected by laser and stereo vision sensors. Due to different properties of the sensors the fused objects have more details and give better environment view. We used Bayesian fusion at this level to get correct object positions and classification information to the laser objects to get better tracking results.

Track level fusion is useful when there are many false alarms and we want to confirm real tracks rapidly.

We have seen that in most of the cases object level fusion is more appropriate then the other two levels because low level is too early and many sensor specific information are lost due to finding a common representation scheme quite early in the process and track level is too late that sensor uncertainty differences may have already resulted into many false positives and miss detections. At object level appropriate data association decisions can be made using knowledge about sensor properties. However for much noisy sensors track level fusion may give better results.

Last important work detailed in this chapter is the fast detection of road borders from laser data and then using this knowledge to remove false moving objects detected near road boundaries due to sensor noise. Without requiring to use another sensor this technique not only reduces false alarms but also gives more institutive road shape.

Although people have tried fusion between laser and stereo vision in different ways however we believe that the fast and simple fusion techniques that we have developed and detailed in this chapter are very useful in similar scenarios mentioned in this chapter. Especially object level fusion level is relatively more appropriate level as low level is too early and track level is too late. However both low and track levels can be useful in certain situations as reasoned above.

CHAPTER 4

Laser and Mono Vision Fusion

Monocular cameras are one of the first sensors to be used in research for building intelligent vehicles. As noted in first chapter, both *Starnford Cart* and Japan's Tsukuba Mechanical Engineering Lab vehicles, the very first examples of intelligent vehicles, used mono vision as main sensors. Laser sensors, because of their ability to give precise distance information, and mono vision being able to give content rich information, are now sensors of choice for intelligent vehicle perception.

In this chapter, we detail our work of fusion between laser and monocular vision camera. The main difference of this work w.r.t our fusion work between laser and stereo vision, detailed in previous chapter, is that both stereo vision and laser are compatible sensors with some aspects of each sensor better than the other one. So both were given equal importance for fusion at different levels. However, mono vision is not compatible with laser scanner and each of them gives details of different aspects of the environment. So in this case they complement each other and information from mono vision are used to add more details to the environment model constructed by laser scanner. Laser is used as primary sensor whereas mono vision as secondary sensor. An important implication of this setup is that failure of laser scanner will cause a complete failure of the perception system.

We have performed fusion between laser and mono vision at two levels: low and objects detection level. We see that track level fusion between these sensors adds more overhead than advantage. Essentially track level fusion reduces to object level fusion but by taking extra processing resources. Since there are no depth information available from mono vision so tracking a moving object in the images will add no more information for the laser tracks other than class information. Same results are achieved by fusion at object level.

Rest of this chapter is structured as follows: Next section details fusion between laser

and mono vision at low level. Here again we use occupancy grid framework to do the fusion, using information from laser we find corresponding regions in the image and using background subtraction technique we build an occupancy grid for vision, for laser we use same technique as explained in previous chapter to build occupancy grid. These two grids are combined using a fusion technique to get fused grid. Experiments performed on a simulator test vehicle show that this technique is effective and can be used in certain environments. Section 4.2 details our object level fusion between these two sensors. Here again, using laser scanner as primary sensor we extract moving objects from laser data and then using camera calibration information object rectangles are projected on the image to get regions of interest in them. These regions of interest are passed to classifiers to get the class information for each laser object. Using these classification information we do model based tracking of laser objects to get smooth tracks. Although we have used this model based tracking concept here with laser and mono vision, however it can also be used easily with stereo vision and laser scanner case because it is based on classification information of the objects. Last section of this chapter gives a summary and some discussion of the techniques presented here.

4.1 Low level fusion

In this section we explain our low level fusion work between laser and monocular color camera (architecture shown in figure 4.1). This work is detailed in our publication Baig and Aycard (2010). Like our previous low level fusion work between laser and stereo vision it is also based on occupancy grid frame work. Main novelty of this work comes from the fast inverse sensor model that we have developed for mono vision to construct a low level occupancy grid. For each new data frame from these sensors we construct an occupancy grid for both of them and then fuse these grids to construct a fused occupancy grid. Although the laser processing is similar to that in last chapter however the inverse sensor model for mono vision is completely different from that of stereo vision.

Since this work uses a color camera so it uses a new experimental setup as explained in next sub section. Then we summarize our laser processing to build laser grid and detail vision processing and vision sensor model based on background subtraction technique. Fusion and some results of experiments conducted on simulated demonstrator vehicle are presented at the end of this section.

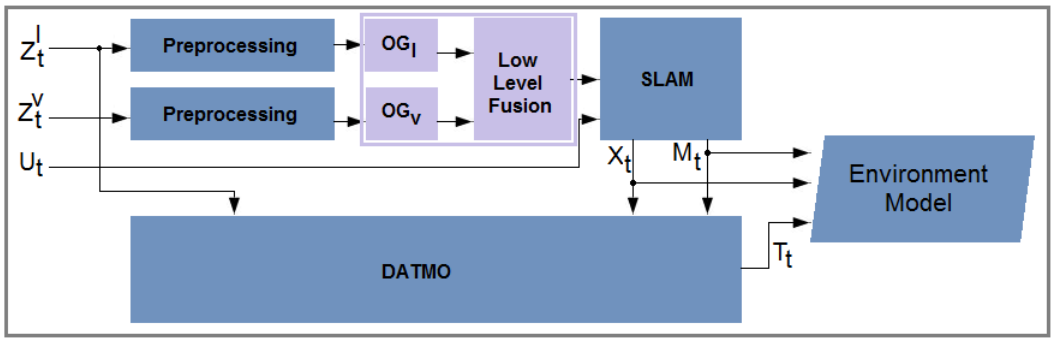

Figure 4.1: Low level fusion with perception architecture

4.1.1 Experimental Setup

The demonstrator vehicle used to get data sets for this work is a simulator of a Cycab[1] vehicle prepared by INRIA. This simulator provides a graphical interface along with a movable Cycab car. This Cycab car has simulated laser and camera sensors fixed on it. We can also load different static and moving objects in the view by executing their respective scripts. With this simulator creation of experimental environment becomes very easy.

The figure 4.2 provides a close view of the Cycab vehicle and the two sensors mounted on it. In the simulator environment this Cycab vehicle can move around in the parking.

The simulated laser scanner has a field of view of 180° and a maximum range of 90 meters. It has 360 laser beams called channels.

4.1.2 Occupancy Grid

As explained in last chapter occupancy grid is a multi-dimensional tessellation of space into cells where each cell stores a probabilistic estimate of its state. To be consistent with our previous notations we represent the output of laser as z_t^l and output from camera on the Cycab simulator as z_t^v, at time instant t. Here again laser output consists of N tuples ($z_t^l = \{(r_1, \theta_1), \ldots, (r_N, \theta_N)\}$) for N laser hit points. However the output of camera is a raw color image at time instant t.

The general theory of constructing an occupancy grid is the same as described in section 3.1. Cell update probability is represented as $P(m|x_{1:t}, z_{1:t})$ and, using log odds form, was calculated (in section 3.1) to be:

$$\log Odds(m|x_{1:t}, z_{1:t}) = \log Odds(m|z_t, x_t) + \log Odds(m|x_{1:t-1}, z_{1:t-1}) \tag{4.1.1}$$

[1] http://cycabtk.gforge.inria.fr/wiki/doku.php?id=download

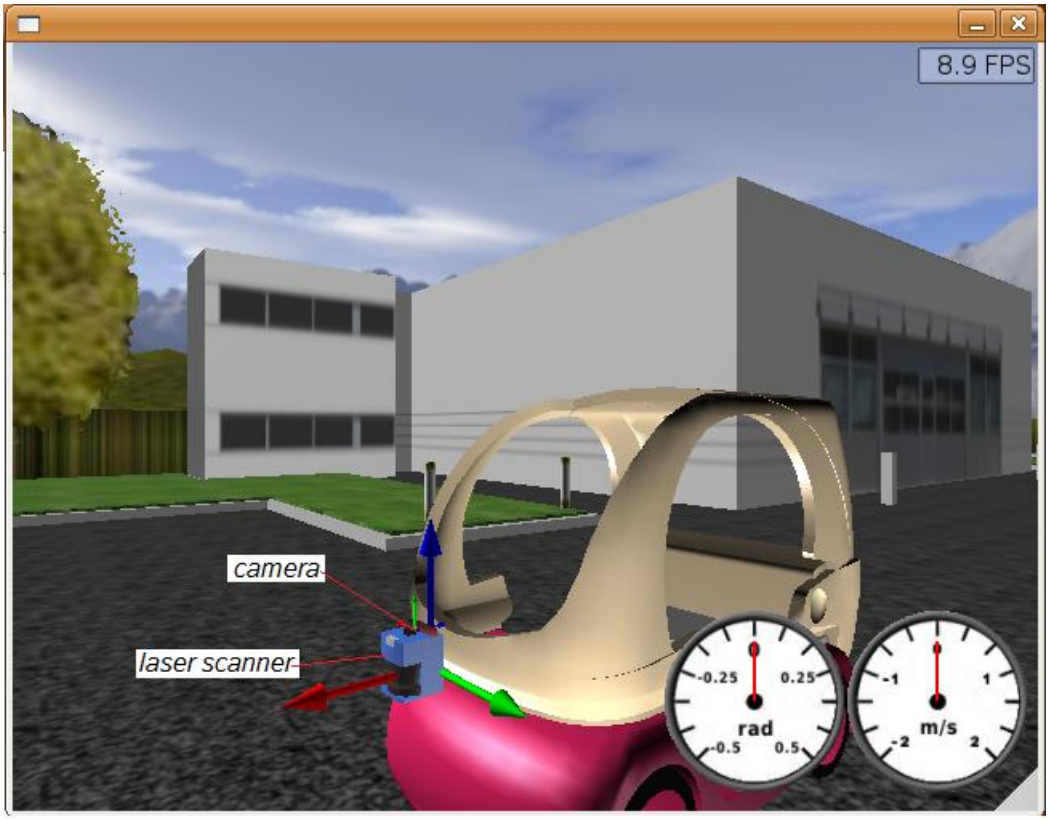

Figure 4.2: Cycab with laser scanner and camera mounted on it.

So giving a brief summary of occupancy grid construction for laser in the following section we explain our vision processing and inverse sensor model used to construct occupancy grid for camera sensor in the section next to the following.

4.1.3 Occupancy Grid for Laser

Using the inverse sensor model shown in figure 3.4 we use the same process for constructing occupancy grid for laser scanner as explained in section 3.2.1 of last chapter. Summarizing, occupancy state of a cell is decided by the nearest beam from the center of this cell. If the nearest laser beam goes beyond this cell it has low probability of occupancy (signifying empty state). However, if the nearest laser beam terminates in the cell or quite close to this cell, it has high value of occupancy. Otherwise cell has a value of 0.5 indicating that the occupancy state is unknown.

4.1.4 Vision Processing

As we know that an image from a camera is a 2D projection of a 3D view. Whereas a laser scanner scans semi-circle shaped horizontal plane at a fixed height from ground.

Since our target is to construct two grids, one for laser and one for camera, and then to merge them to get a combined occupancy state of the view. Therefore it is very important that constructed grids correspond to the same area of the 3D view for both sensors. Horizontal plan scanned by laser corresponds to a sort of horizontal strip in the image. This implies that if we want to calculate occupancy grid for camera then we, first of all, need to find the points in the image corresponding to each laser beam. This requires us to perform transformations from laser reference frame towards camera reference frame. These transformations will give us pixels in the image that correspond to the hit points of the laser beams. So for each laser hit point $P = [x, y, z, 1]^T$ we need the pixel $X = [u, v, 1]^T$. For laser we have $x = r\cos(\theta)$,$y = r\sin(\theta)$ and $z = 0$ since our global origin is same as that of laser reference frame's origin. We use following equations from vision literature to calculate these transformations.

$$X_{3\times1} \sim PM_{3\times4} \times P_{4\times1} \tag{4.1.2}$$

Where Projection Matrix PM is given as:

$$PM_{3\times4} = K_{3\times4} \times \begin{pmatrix} R & -Rt \\ 0^T & 1 \end{pmatrix}_{4\times4} \tag{4.1.3}$$

Here K is the camera calibration matrix. R is rotation matrix and t is the camera translation vector. Camera is calibrated and fixed with respect to laser scanner so we know values of these matrices and vector.

By applying these equations we can calculate image pixels for each laser beam if its termination point lies in the camera view. Now in order to classify these pixels as belonging to objects or background, we need a technique to differentiate background from the foreground. This technique is explained below.

Object Detection

Our object detection technique is based on background subtraction. We classify a pixel as either belonging to background or foreground. In order to apply this technique we need to learn background first. We have used multivariate Gaussian distribution technique to learn background. In our case bounding fence of the parking area forms the background. Learning background essentially means learning mean vector (μ) and covariance matrix(Σ) of the multivariate Gaussian distribution. For this purpose we collect different sets of pixels belonging to background from different distances by moving cycab in the parking area and manually segmenting the regions belonging to

background. Figure 4.3 shows the simulated parking area with bounding fence and parked cars in it. From training data we can calculate the μ and Σ parameters of the multivariate Gaussian distribution for N background pixels $X = [r, g, b]^T$ as follows:

$$\mu = \frac{1}{n}\sum_{i=1}^{N} X_i \tag{4.1.4}$$

$$\Sigma = \frac{1}{n}\sum_{i=1}^{N} (X_i - \mu).(X_i - \mu)^\top \tag{4.1.5}$$

The advantage of this technique is that object detection will work even if the simulator cycab is moving in the parking area.

After we have learnt background parameters we can apply following equation to classify any pixel $x = [r, g, b]^T$ as belonging to object or background.

$$f_x(r, g, b) = \frac{1}{(2\pi)^{3/2}|\Sigma|^{1/2}} \exp\left(-\frac{1}{2}(x - \mu)^\top \Sigma^{-1}(x - \mu)\right) \tag{4.1.6}$$

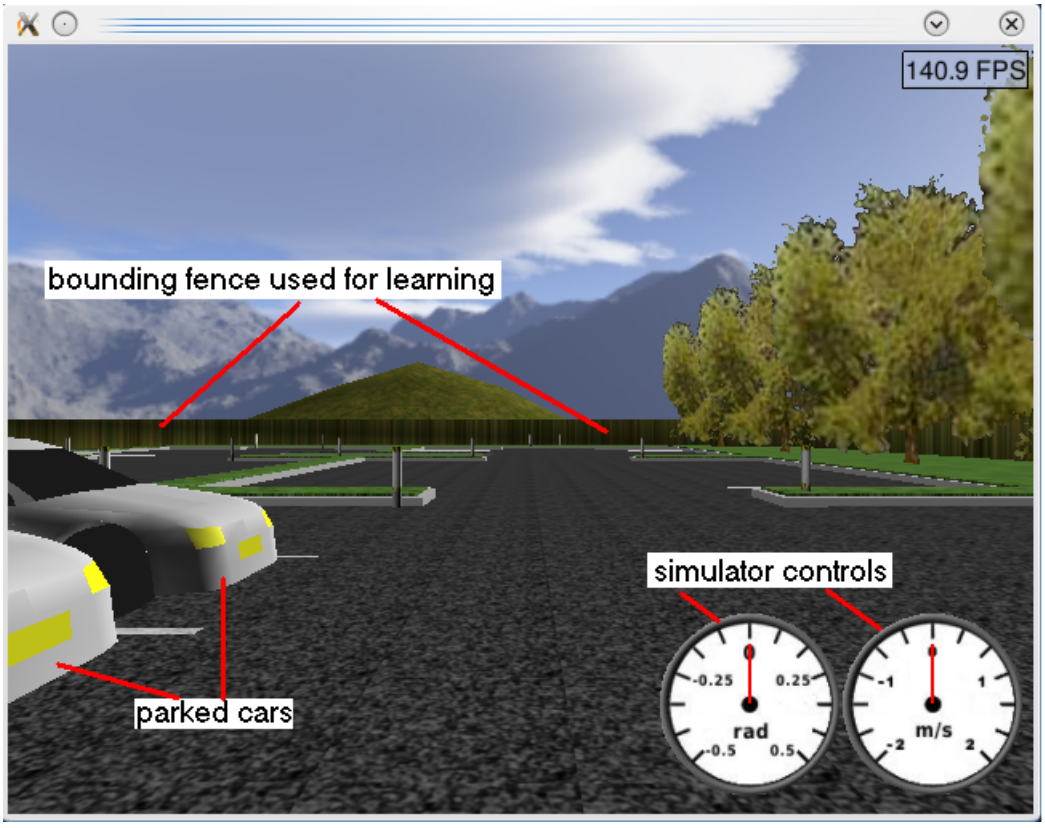

Figure 4.3: Simulated parking area.

If the probability is smaller than a threshold value, learnt experimentally, then the pixel belongs to an object otherwise it belongs to background.

Occupancy Grid For Camera

For camera measurement is defined as $z_t = \{z_t^1, ..., z_t^k\}$ for k individual pixels corresponding to k laser beams' termination points visible in the camera view. Whereas one pixel measurement is defined as $z = [c]$ where c is the class of the pixel. For constructing an occupancy grid we need to define inverse sensor model for camera to calculate $P(m \mid z_t, x_t)$ term in equation 4.1.1. We define inverse sensor model as follows, if a pixel is classified as belonging to background then all the cells lying along a line from camera to the end of the grid have low value of occupancy. But if the pixel is classified as object then all the cells lying along the line have relatively high value of the occupancy state. Since we do not have depth information from the images so all the cells along the line have high value of occupancy if the pixel is classified as belonging to object. The cells lying outside of the camera view have a probability value of 0.5 which means unknown state.

The steps to construct an occupancy grid for camera are as follows:

- Find pixels of the image corresponding to laser hit points for those laser beams which lie in the camera view. We need this step only to know the image region that is common with laser so that both occupancy grids are compatible.
- Find class of these pixels to see which of them belong to background and which of them belong to objects.
- Using definition of inverse sensor model given above, set probabilities of all cells using pixel classification information.

Figure 4.4 shows an occupancy grid constructed from camera with laser hit points projected on this grid. We can see that most of the points lie in the area where there is high probability of occupancy, showing the effectiveness of this technique.

4.1.5 Fusion

Fusion is the process of combining information from multiple sources in such a way that the combined information are more useful in some sense. Our objective is to combine information from laser and camera sensors. An advantage of fusion at this level is that there is no information loss due to any interpretation of data which is necessary for higher levels.

There exist different fusion techniques of which we have used weighted linear combi-

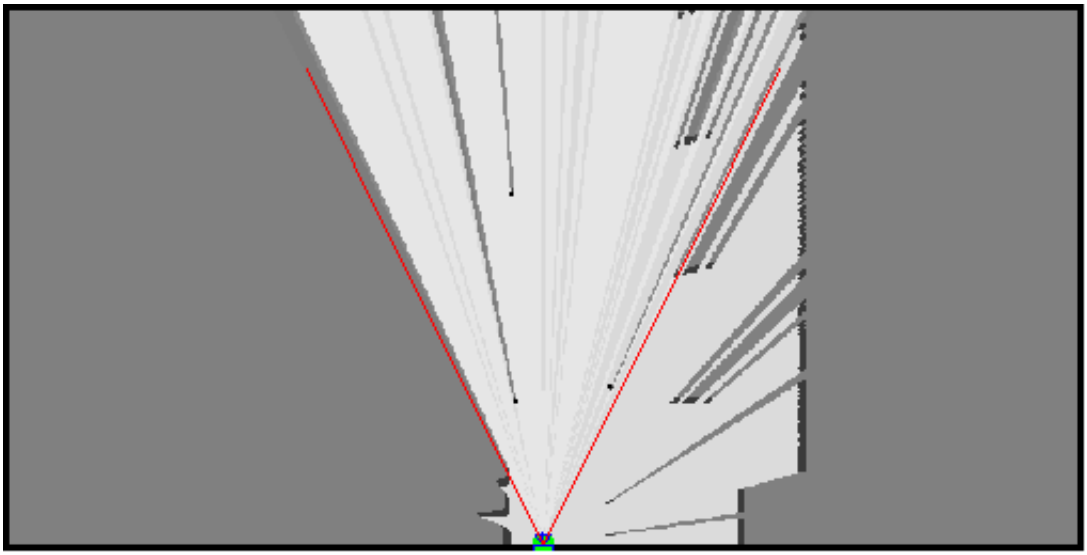

Figure 4.4: Occupancy grid for camera within red lines with black dots representing objects detected by laser

nation of the two grids. Fusion equation is given below:

$$FusedGrid_{t,i} = \alpha.CameraGrid_{t,i} + (1 - \alpha).LaserGrid_{t,i} \tag{4.1.7}$$

Here α is the weighting factor for camera and its value is between 0 and 1.

Another fusion technique that we have used is the conservative estimate. Let m_i^l is the i^{th} cell of the occupancy map built for laser sensor and m_i^c is the same cell for camera sensor. Then the value of same cell in the fused grid is given by:

$$m_i = max(m_i^l, m_i^c) \tag{4.1.8}$$

This map is the most pessimistic map given its components: If any of the sensor-specific map shows that a grid cell is occupied, so will the combined map. While this combined estimator is biased in factor of occupied maps, it is more appropriate than the Bayes filter approach when sensors with different characteristics are fused Thrun *et al.* (2005).

As stated in previous chapter, the main task is to construct two compatible grids by defining appropriate inverse sensor models for all sensors. After that we have the grids then different ways, that depend on the characteristics of the sensors, can be easily devised to combine these grids.

4.1.6 Results

Fusion results of weighted linear combination for three different values of α are shown in figures 4.5, 4.7 and 4.8. The red lines in the fused view show the camera view with

respect to laser view. The respective scenes for these results are shown in figures 4.6 and 4.9. Some important information can be seen easily from these results: first, the occupancy grid constructed for camera is compatible with that of laser and belong to the same part of the environment. Second, mostly an object seen by laser is also seen by vision on the same place, although there are no depth information present for vision, however, the free space seen by both sensors is same giving more weight to this part of the grid.

Similarly the figure 4.10 shows fusion results for conservative approach for the same scene as shown in figure 4.9. From these results we see that other than depth information for vision, occupancy grids constructed for both sensors are similar and object seen by laser sensor is also detected by vision.

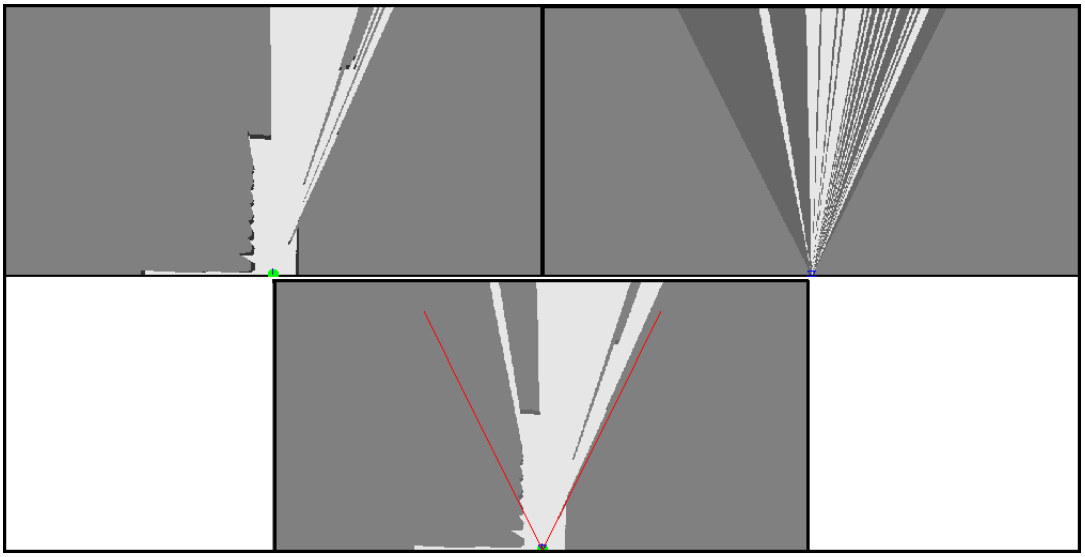

Figure 4.5: Laser OG at left, Camera OG at right and Fused OG at bottom for $\alpha = 0.1$. Scenario shown in figure 4.6.

We can see from these results that the two occupancy grids constructed for laser and camera are very good approximation of the actual environment. And the fused view shows the combined information of both the sensors.

If depth information are used from laser scanner for objects detected by vision then both grids will become compatible and will give even better view of the external environment. This fusion is useful for structured indoor environment where background color can be separated and where we need to be more sure about the free space before the robot.

An important other use of vision processing described in this section is that we can detect objects easily and rapidly in the image. The consecutive pixels detected as belong

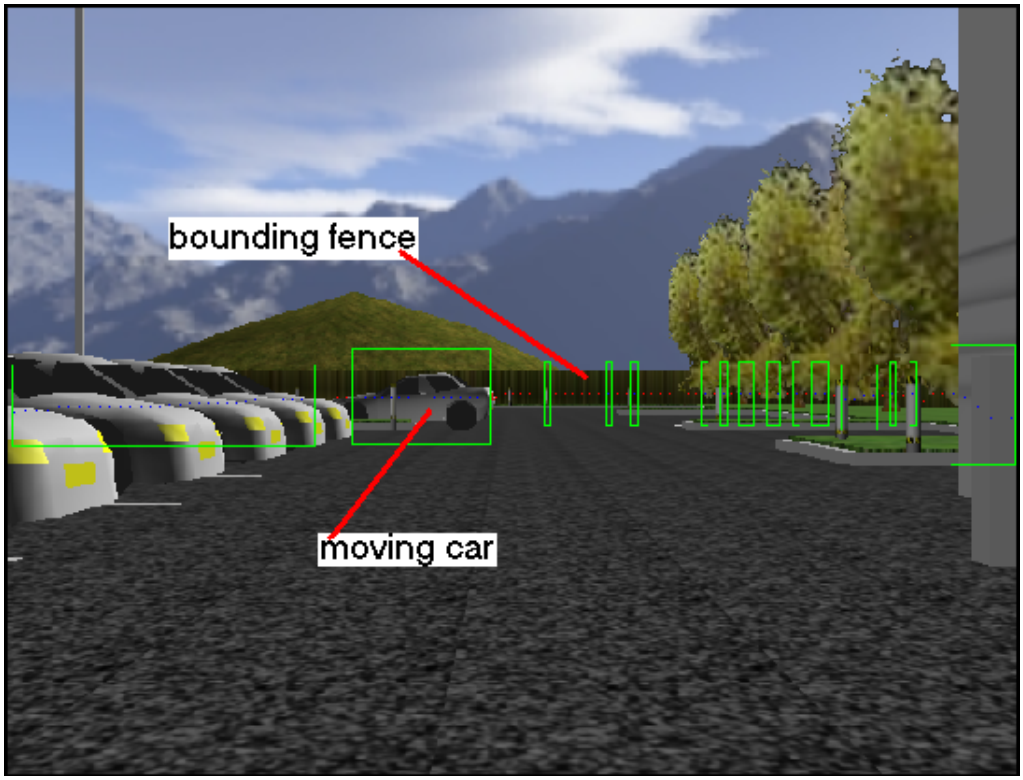

Figure 4.6: Scene for $\alpha = 0.1$

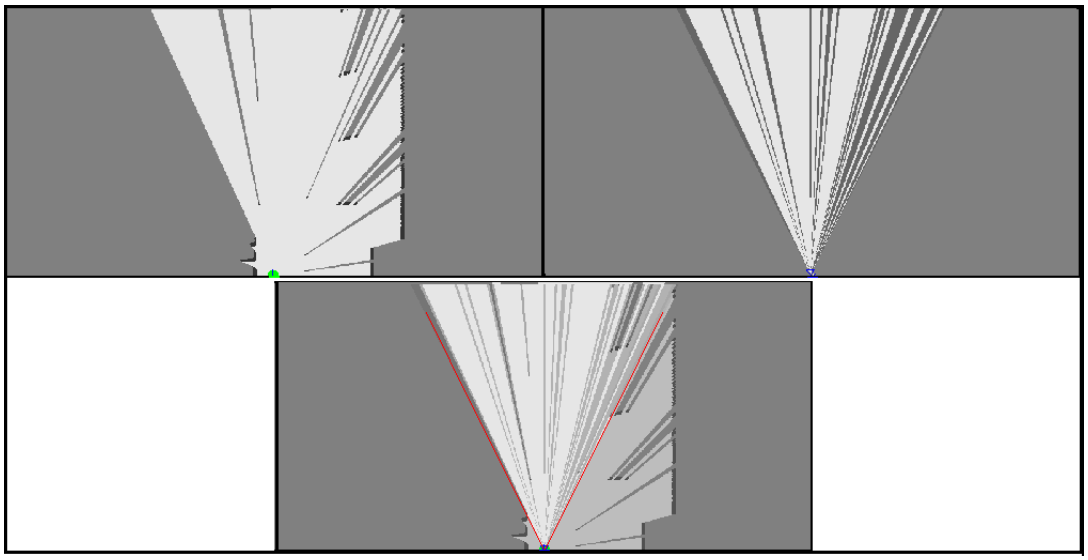

Figure 4.7: Laser OG at left, Camera OG at right and Fused OG at bottom for $\alpha = 0.4$. Scenario shown in figure 4.9.

to foreground define the width of the object, a similar search can be done vertically for each detected object to find the height of each object. The results of object detection in the images are shown in figures 4.6 and 4.9 using green rectangles.

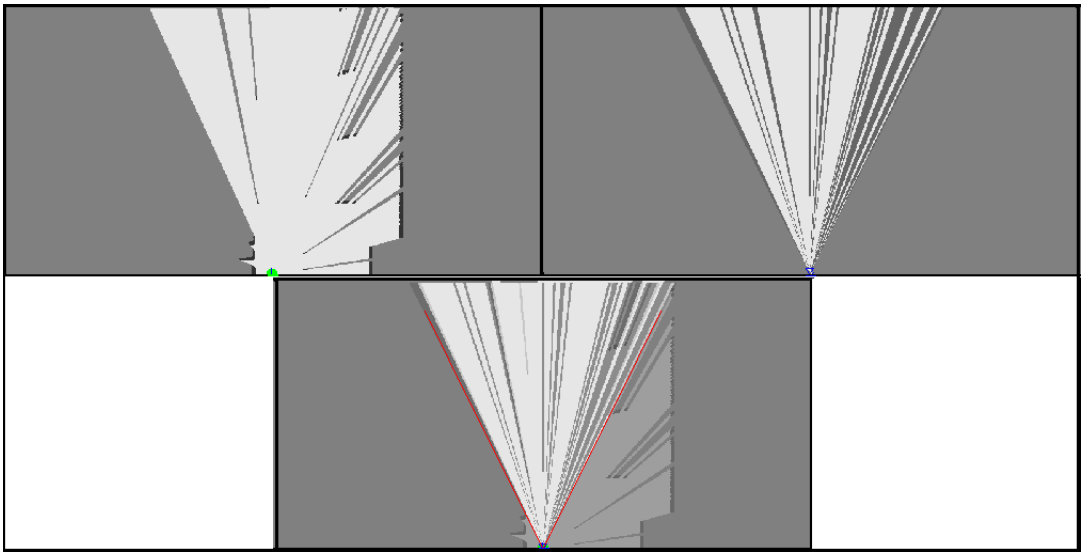

Figure 4.8: Laser OG at left, Camera OG at right and Fused OG at bottom for $\alpha = 0.7$. Scenario shown in figure 4.9.

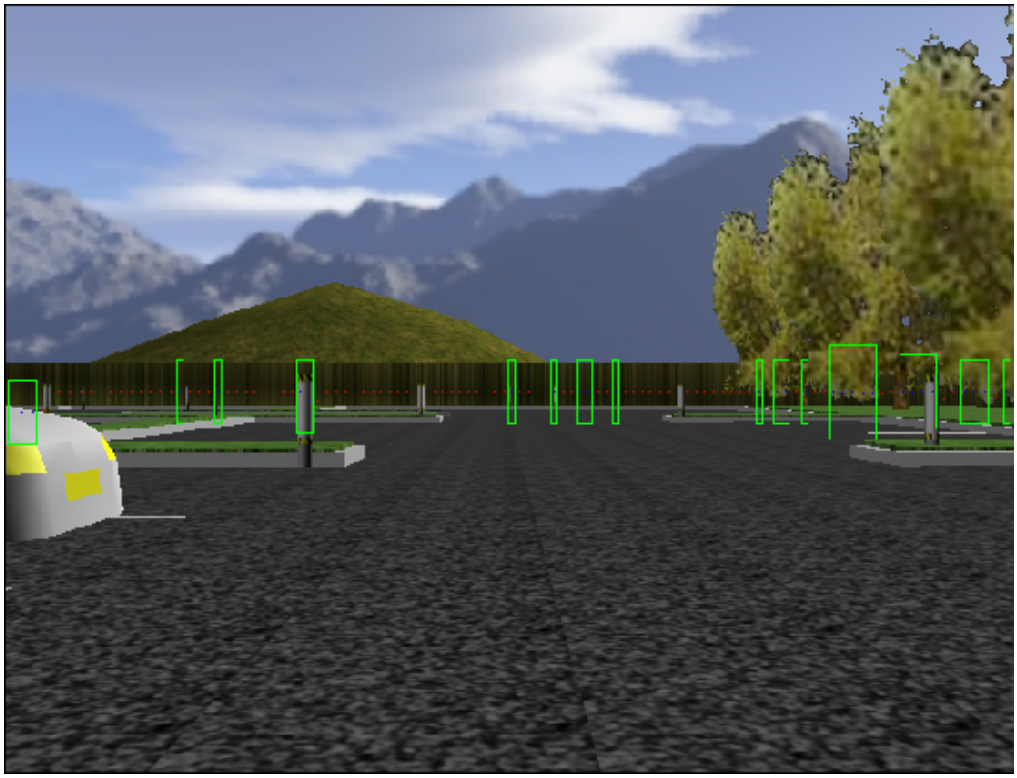

Figure 4.9: Scene for $\alpha = 0.4$ and $\alpha = 0.7$

4.2 Object level fusion

In this section we present object level fusion between laser and mono vision. Here laser is used as primary sensor and vision is used to get classification information for laser

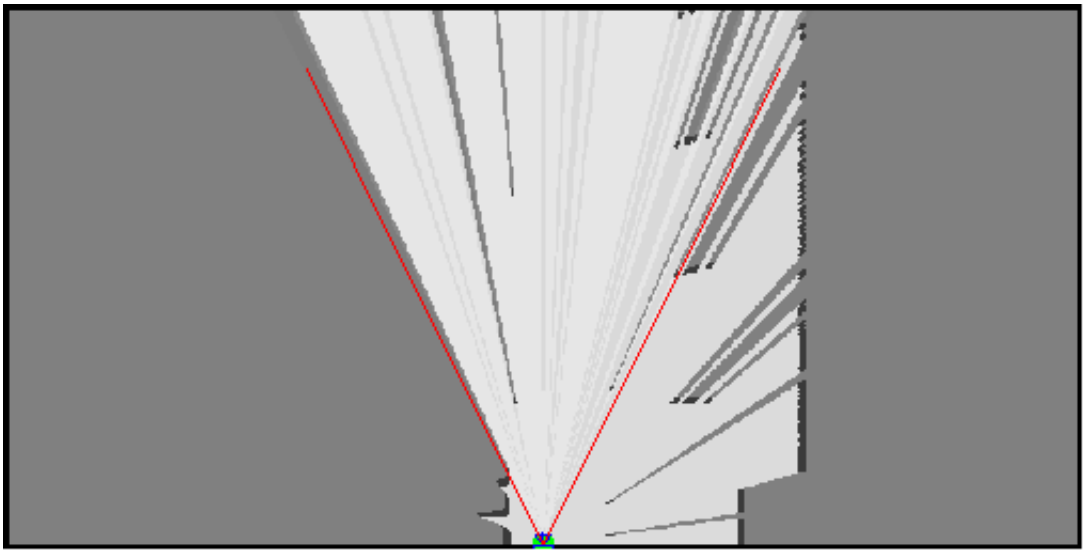

Figure 4.10: Fusion result for conservative approach. Scenario shown in figure 4.9.

objects. To achieve this, laser data is processed to solve *SLAM* and *DATMO* problems as explained in previous chapter. To get classification information of detected moving objects, laser points belonging to each object are projected on the image to get an position of the object on the image. An appropriate rectangular region is extracted from the image for each object and passed through a classifier to get class information. These information are used to do model based tracking as explained below.

An important problem of object tracking using laser scanner is the smoothness of the trajectory of the moving object. Even if the object is detected and tracked successfully the shape of the object path is usually not smooth due to different shapes of the object segment detected in different data frames. This problem and its solution using object class information are shown in figure 4.11 (from (Vu and Aycard, 2009)).

To solve this problem for laser scanner a model based object tracking approach was suggested by Petrovskaya and Thrun (2009) where they used a flexible box model to represent cars and introduced a method to learn object sizes during tracking. Later on a similar approach was used by Vu (2009), however he formulated the detection and tracking of moving objects as a single simultaneous problem and model detection of the moving object was incorporated into this simultaneous problem. This approach results into a large dimensional solution space, where solution is a set of tracks observed in the data frames over a sliding window of T frames. They used an MCMC based sampling technique to find solution from this space .

Although above methods may work with high resolution laser scanners they are not suitable for noisy and low resolution laser sensor. In this scenario we have used a gray

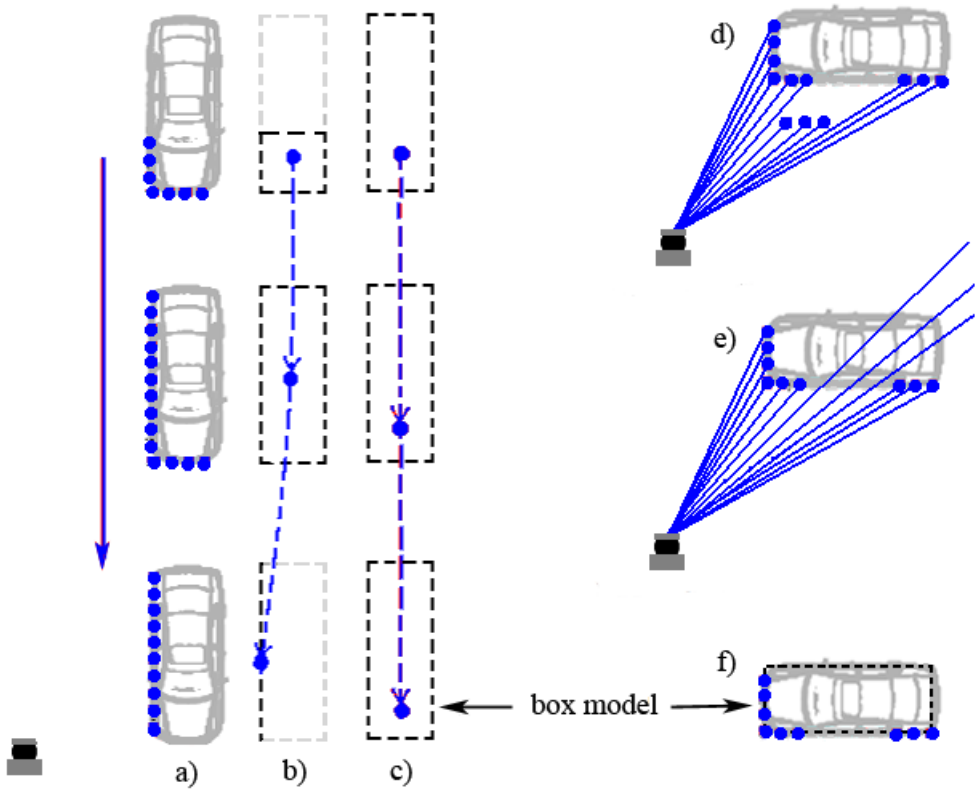

Figure 4.11: Known problems with laser-based detection and tracking. (a) an example of a car approaching the laser scanner; (b) car tracking using centers of laser impact bounding-boxes leads to incorrect result; (c) correct tracking using car model; (d)(e) objects can be divided into several segments making tracking harder that requires object merging, track grouping; (f) using car model can overcome these problems.

scale monocular vision camera along with laser scanner to do model based tracking. Laser data is used to find moving objects, the laser points belonging to these moving objects are projected on the image to find regions of interest in the image, these regions are passed to a classifier to know the class of the objects in these regions, based on this output a model is fit onto the laser data during tracking process to get smooth track. This process is detailed in following subsections.

4.2.1 Data Set

Data set for this work comes from Volkswagen demonstrator described in previous chapter and consists of a laser scanner with 2° resolution and images from a gray scale monocular camera (left camera of the stereo vision system mentioned in previous chapter serves as the monocular camera). As usual we represent the current scan of laser

data as z_t^l and current image from the camera as z_t^v.

4.2.2 Moving objects detection from laser data

The process of detecting moving objects from laser data is exactly same as described in section 3.3.1 of last chapter. As detailed there, the moving objects are detected by observing inconsistencies between the latest map of the environment generated by *SLAM* module and current laser data frame. The laser hits in z_t^l frame seen in empty areas of map belong to moving objects, and a segmentation process is used to make objects from individual laser hit points. So a moving object is a set of points classified as moving and belonging to a segment: $O_i = \{P|P \text{ is moving and belongs to } i\text{th segment}\}$, here P is a laser hit point.

4.2.3 Fusion: finding region of interest and its class for a dynamic object

Finding region of interest for each moving object in the image is directed by the laser points comprising this object. The distance between two extreme points in the object segment can be used as a guide for object rectangle in the image. However to take the laser data problems into account (like the ones shown in *d* and *e* parts of figure 4.11) we actually define three rectangles for the three types of objects (vehicle, bike and person) that we have handled in this work. These rectangles are defined as follows: for each segment we first find its "center of gravity" (a point having the mean values of x and y coordinates of the segment points) and then one rectangle for each type of the object is defined around this point (widths and heights are different for these classes). The four corners of these rectangles are projected on the image using camera calibration information to find three regions of interest. These regions of interest are passed to their respective classifiers and the one returning the highest confidence is taken as the class of this object. Due to the distance given by laser the perspective ratios of the width and height for each region of interest are automatically taken into account (an object farther away will have small rectangle whereas an object closer to the vehicle will have larger rectangle).

Since in this thesis we were primarily concerned with laser data processing and development of fusion techniques, so we used vision processing done by other teams. The same idea applies to this fusion level and we want to use open source classifiers for objects classification. However, so far we have not found some good classifiers for vehicles and cyclists because as stated by Gupte *et al.* (2002) and Pinto *et al.* (2008), this is a hard problem. A good literature review on different techniques of object classification

in camera images is discussed in (Deselaers *et al.*, 2010, Filipovych and Ribeiro, 2007). For this work we have used some hand labeled data. For each data set we do a labeling run in which each object is projected on the image to see its class and this class is saved in a file for each object. In second run class of each object is read from this file for model based tracking.

4.2.4 Model based tracking

As detailed in section 3.3.4 of chapter 3, we have implemented a tracking scheme that uses MHT for data association and IMM as filtering schemes. However before the tracking process we best fit a model (currently we are using only three models, a rectangle for vehicles, a small rectangle for cyclists and a dot circle for pedestrians) to the laser points of each object segment, the dimensions of this rectangle depend on the class of the object determined in the previous step. The center of this rectangle is taken as the position of the object. During data association step, hypotheses involving different types of objects (objects having different class types) are ignored resulting in reduced number of hypotheses. If the shape of the laser points is such that a model can fit in more than one directions then we use information from its previous position to determine its correct direction. For very first occurrence of the object we select one randomly from possible directions and hope that if its not correct then it will be corrected in next iteration when we will have new position information of the same object.

Although for current work we have used hand labeled data, however, by introducing some noise (wrong class information) for some objects we have seen that this effect of errors reduces greatly in next data frames when new position information become available and some knowledge of shape is used from laser data.

4.2.5 Results

Tracking results of a pedestrian are shown in figure 4.12. Figure 4.13 shows another useful aspect of this fusion related to HMI (human machine interface) presented to the driver to show information in more intuitive way. Although showing moving objects as dots or rectangles is common however if system is able to infer class information of the moving targets then it is far better to show class icons on the screen than showing dots or rectangles.

Although it is relatively difficult with low resolution laser scanners, however if high resolution scanners are used then class of the object can be, to some extent, inferred from the shape of the object segment given by laser scanner.

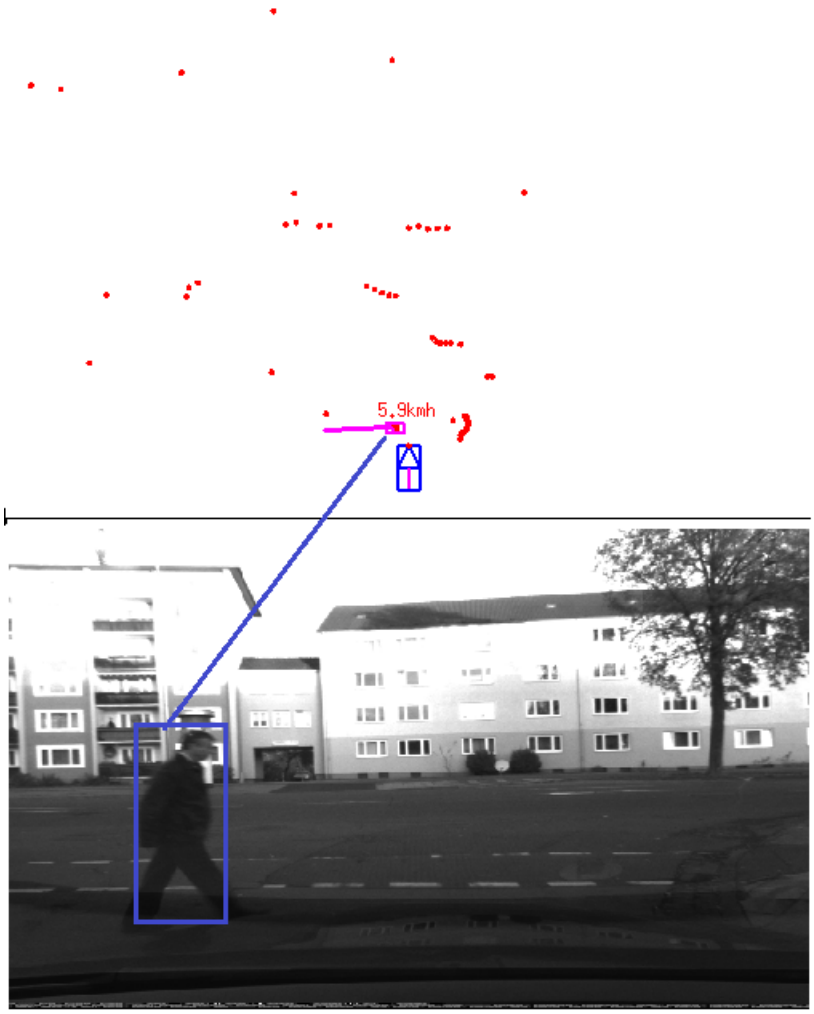

Figure 4.12: Tracking results of a pedestrian. Red dots show current laser scan points on static objects.

In general we see that as vision processing advances and fast and reliable classifiers are built then this fusion can be very much useful for driver assistance systems. True class information in HMI systems results in more driver satisfaction than using moving dots or rectangles.

4.3 Synthesis

In this chapter we have detailed our fusion work between laser and monocular vision camera at two levels: low and object level. Based on occupancy grid framework low level involves constructing two compatible grids, one for each sensor and then

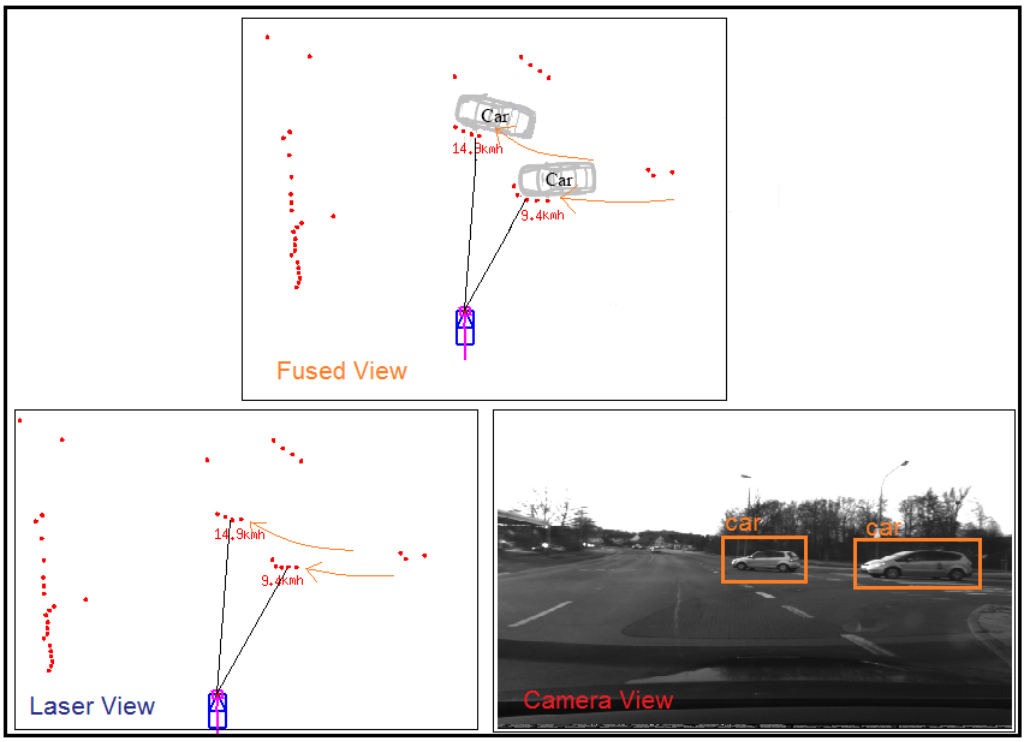

Figure 4.13: Fusion example between laser and mono vision. Laser (bottom left) detects moving objects, calculates their position and velocity information whereas vision (bottom right) classifies them as cars. After fusion (top), system knows that it is tracking two cars.

combining these grids to get a fused grid. For vision our technique was based on classifying image pixels corresponding to laser hit points as belonging to background or foreground. A pixel belonging to background means that there is no object before vehicle in this direction. But if a pixel is classified as foreground then it means that there is an object before the vehicle in this direction. Fusion between two grids was performed using different weights for two grids. This fusion is more useful in the scenario where we need more confidence about empty space in front of the vehicle, for example for applications that need to know the drivable area. The weak point of this approach is that it is only useful for indoor environments where background can be distinguished from foreground.

Fusion at object level involved finding regions of interest in the image for each dynamic segment detected by laser scanner and then using a classifier to find the class of the object of this region. The knowledge of the classification information is used in next step to do model based tracking. The main objective of the model based tracking is to get smooth trajectories of moving objects which is not possible from laser data alone, since same object appears to have different shapes when it is at different positions relative to

the sensor.

From both of these fusion strategies it can be easily seen that laser is taken as primary sensor whereas mono camera is used as secondary sensor where information from camera are used to support information from laser to get better environment understanding. This primary and secondary relationship between laser and camera further helps to avoid processing whole image. In both cases we only process specific pixels (or regions of interest) to get required information making the whole process very fast.

In general fusion between laser and mono vision is used to convey more information to the driver in HMI components by displaying intuitive icons and images while displaying other information on the screen. These information make the driving experience more comfortable and relaxing.

CHAPTER 5

Application: Safety at Intersections (INTERSAFE-2)

Our fusion work especially at object level between laser and stereo vision was integrated into Volkswagen demonstrator used for INTERSAFE-2[1] project. INTERSAFE-2 is an *FP7* project funded by *European Commission* and deals with safety at intersections. In this chapter we summarize software architecture of INTERSAFE-2 safety system and briefly mention contributions of different teams which collaborated on different modules of this project for Volkswagen demonstrator. Our this work is published here (Aycard *et al.*, 2011).

Although most of the so called black spots have been eliminated from the road networks, however this is not true for the intersections, they can still be regarded as black spots. According to the data collected for this project, depending on the region and country, from 30 to 60% of all injury accidents and up to one third of the fatalities occur at intersections due to the complex scenarios there. This project aimed at the development and demonstration of a Cooperative Intersection Safety System (CISS) that is able to significantly reduce injury and fatal accidents at intersections.

The consortium for this project consisted of twelve partners[2] who worked on three demonstrator vehicles: a car demonstrator from BMW, another car demonstrator from VW and a truck demonstrator from VTEC, they are shown in figure 5.1. Our fusion work was concerned with Volkswagen demonstrator vehicle only.

In this project the concept of equipped infrastructure was also tested, where sensors were mounted on poles in the intersection on different points to get more complete view of the environment and V2I (vehicle to infrastructure) communication was used

[1] http://www.intersafe-2.eu/public

[2] http://www.intersafe-2.eu/public/partners

to convey information to the demonstrator vehicles. With this concept, vehicle was able to see the areas which are not directly visible to the sensors on the vehicle from its current position. Moreover traffic light information were also communicated to the vehicle to decide its action in advance (like, if it needs to apply breaks or there is sufficient time to pass through the intersection without reducing speed). Figure 5.2 shows this concept of on-board (on the vehicle) and infrastructure sensors with their fields of view in different colors.

The objective of the project has been to use information from these on-board and infrastructure sensors to build a complete environment model and to be aware of any moving objects in the environment. Based on these information higher level applications would calculate risk of collision between demonstrator vehicle and any other static or dynamic object on the intersection and would warn the driver or would take appropriate automated action to avoid or mitigate the risk. INTERSAFE-2 has been completed successfully and a public event of its final assessment took place on 17^{th} May 2011 in Wolfsburg Germany.

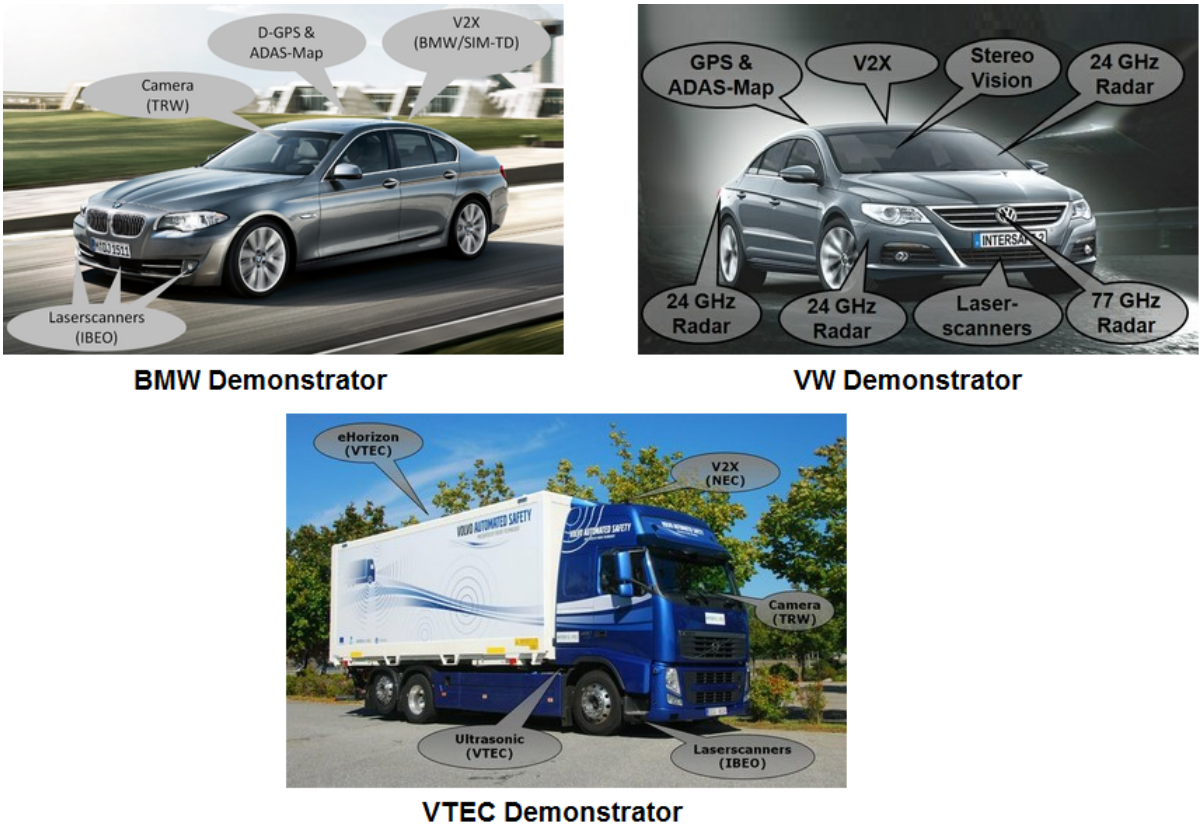

Figure 5.1: Three demonstrator vehicles used for INTERSAFE-2 project with sensors installed on them.

The rest of this chapter is structured as follows: In next section we describe general

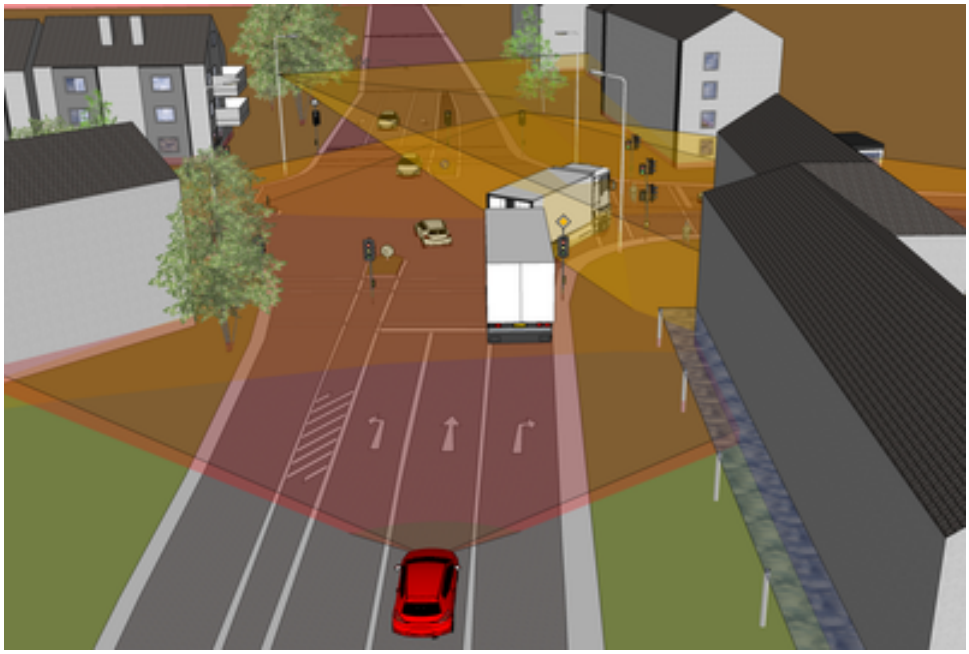

Figure 5.2: Concept of vehicle and infrastructure sensors (best viewed in color).

architecture of the whole INTERSAFE-2 system. Section 5.2 gives details of the system architecture of the host vehicle describing the three layers of this architecture. We summarize the contribution of each partner involved in work on Volkswagen demonstrator in section 5.3. Berief description of integration technology used in the host vehicle is given in section 5.4 and give synthesis of this chapter in last section.

5.1 General architecture

The general architecture of the whole system is shown in the figure 5.3 (from Shooter and Reeve (2009)). INTERSAFE-2 vehicle system is designed to operate in following two modes:

- *Independent mode*, this mode is always active for a host vehicle and makes use of on-board sensors and processing capabilities to observe the surrounding environment. Based on the interpretations made from the sensors data appropriate actions are taken either by warning the driver of the developing dangerous situation or by taking auto actions if the time is too short for a human to intervene. System architecture of this mode consists of three main layers: bottom layer consists of hardware sensors installed on the host vehicle, middle layer called *Perception* is responsible of processing sensors data to build environment model and finding moving objects and calculating their motion attributes. The top layer

called *Application* layer makes use of the environment model and lists of moving objects given by middle layer to assess the environment situations and to take appropriate actions. We will detail this mode in more details in next section.

- *Cooperative mode*, host vehicle works in this mode only when it is on an intersection equipped with infrastructure sensors or there are other vehicles equipped with similar independent mode systems and are willing to share information. Equipped infrastructure has sensors and monitoring devices, information from these components are communicated to the host vehicle via V2I communication. Similarly two equipped vehicles exchange information with each other using V2V communication setup. Both V2I and V2V are collectively called V2X communication setup and consists of necessary hardware and software modules. The main purpose of equipped infrastructures is to enhance the perception and application layer capabilities of an equipped vehicle however this cooperation is optional and host vehicle is always capable of operating in independent mode.

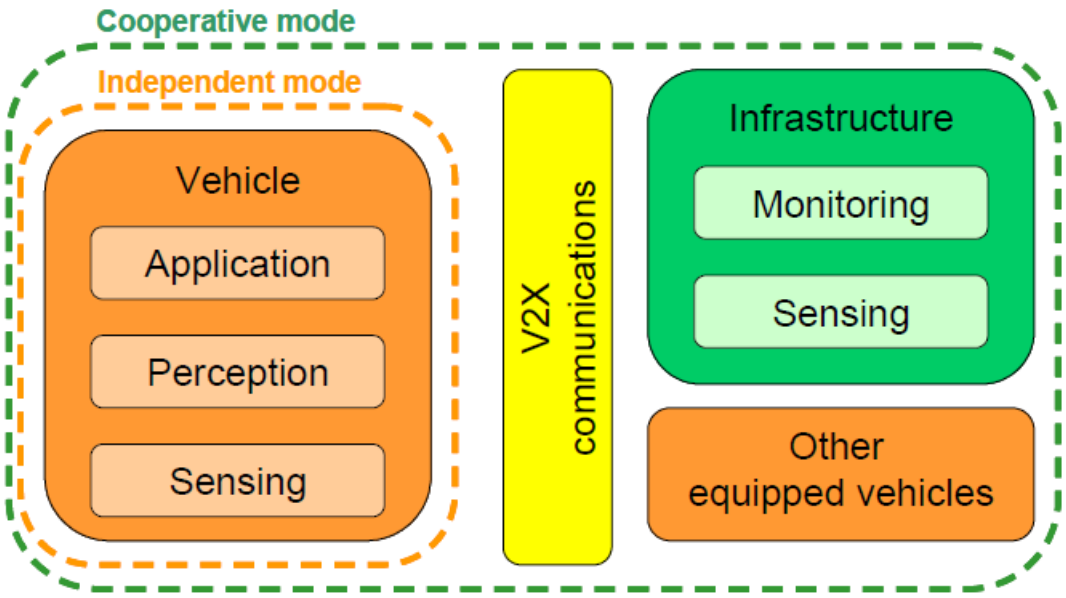

Figure 5.3: Two modes of an INTERSAFE-2 demonstrator vehicle with components of each mode. Independent mode is always active and makes use of on-board sensors to observe environment and respond to any demanding situation. Cooperative mode only works when demonstrator vehicle is on an intersection equipped with infrastructure sensors and monitoring components or is nearby other vehicles equipped with same system and are ready to share information.

Our work in this project was related to the *Perception* layer of the host vehicle working in independent mode. In next section we explain the system architecture of the host

vehicle.

5.2 Host vehicle: system architecture

A generic architecture of an INTERSAFE-2 demonstrator vehicle is shown in figure 5.4 (published in Shooter and Reeve (2009)). This architecture consists of three layers, V2X communication component and maps database. The layers are:

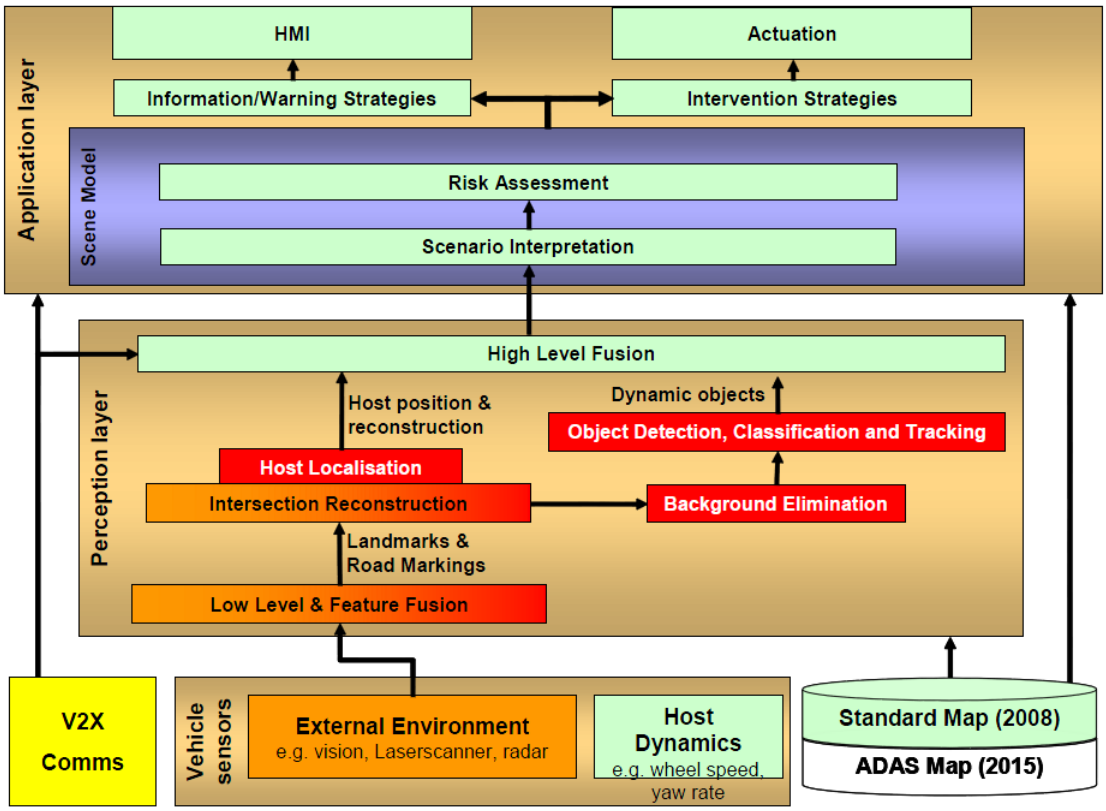

Figure 5.4: System architecture of a demonstrator vehicle.

- *Vehicle sensors layer* consists of hardware sensors installed on the vehicle. These sensors can be divided into two types: external environment sensors and host dynamics sensor. Figure 5.5 shows the types, positions and fields of view of the external environment perception sensors installed on the VW demonstrator vehicle, whereas an Inertial Measurement Unit (IMU) sensor is used to measure host vehicle dynamics. Our work in this project is concerned with the processing of data from Lidar and Stereo vision sensors only.

- *Perception layer* consists of modules that process sensors data to understand external environment. The objectives of processing at this layer include reconstruction

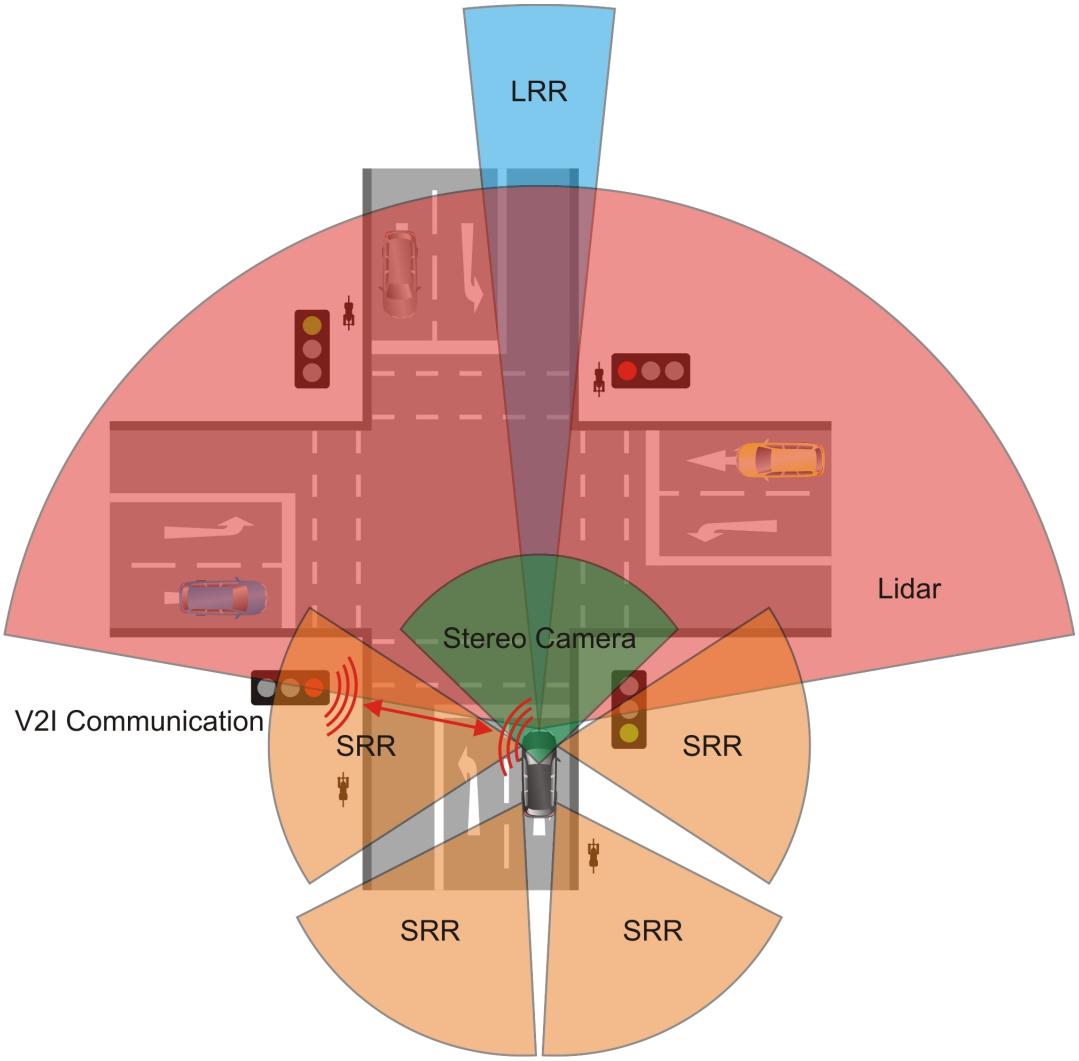

Figure 5.5: External environment perception sensors, their positions and fields of view installed on the VW demonstrator.

of external environment, detection and tracking of moving objects and determining the position of host vehicle in the on line constructed or pre-built maps. Main modules in this layer include fusion at different levels, intersection reconstruction, host vehicle localization and dynamic objects detection, classification and tracking. Our contribution in this project pertains to different modules of this layer.

- *Application layer* uses the environment model and other information provided by perception layer modules for scene modeling. This scene modeling consists of scenario interpretation and doing a risk assessment based on these interpretations. These modules are responsible for finding any developing dangerous situations and taking appropriate actions. The output of this layer consists of

information and warning messages or an active intervention if the interpreted situation is so demanding.

5.3 Partners' contribution

In this project we collaborated with two other teams from Technical University of Cluj-Napoca Romania and INRIA Paris who worked respectively on Stereo vision processing and Risk assessment modules. Stereo vision output consists of a list of classified objects and we perform lidar processing to build environment map and detect moving objects. Detected moving objects are fused with objects given by stereo vision to get list of fused objects. We perform tracking on this list of fused objects to get list of tracks. This list of tracks and the environment model is passed to the risk assessment module to calculate risk of collision between two objects.

In the following we start with stereo vision processing done by team from Technical University of Cluj-Napoca Romania and then summarize our work related to lidar processing, fusion and tracking we end this section with details of risk assessment done by INRIA Paris. The relationship of these modules is shown in figure 5.6. The details of this work can be found in our (Aycard *et al.*, 2011) publication.

5.3.1 Stereo vision processing

Introduction

Dealing with the intersection specific situations imposes substantial changes in the physical setup and in the processing algorithms of a stereo sensor. The use of a wider field of view lenses (shorter focal length) comes at the cost of larger distortions and reduced depth accuracy. For this reason an increased accuracy in calibration and dense stereo reconstruction is required. The use of stereo sensor opens not only the possibility to infer the 3D coordinates for any image point but also the possibility to compute the 3D motion vector for any pixel. The combined use of the 3D position and 3D motion information significantly improves object detection, discrimination between static and dynamic components, and dynamic object tracking.

Stereo sensor architecture for intersection assistance

Based on the requirements of this project a two level architecture of a 6D stereo sensor was proposed (figure 5.7 and figure 5.8). The low level architecture controls the im-

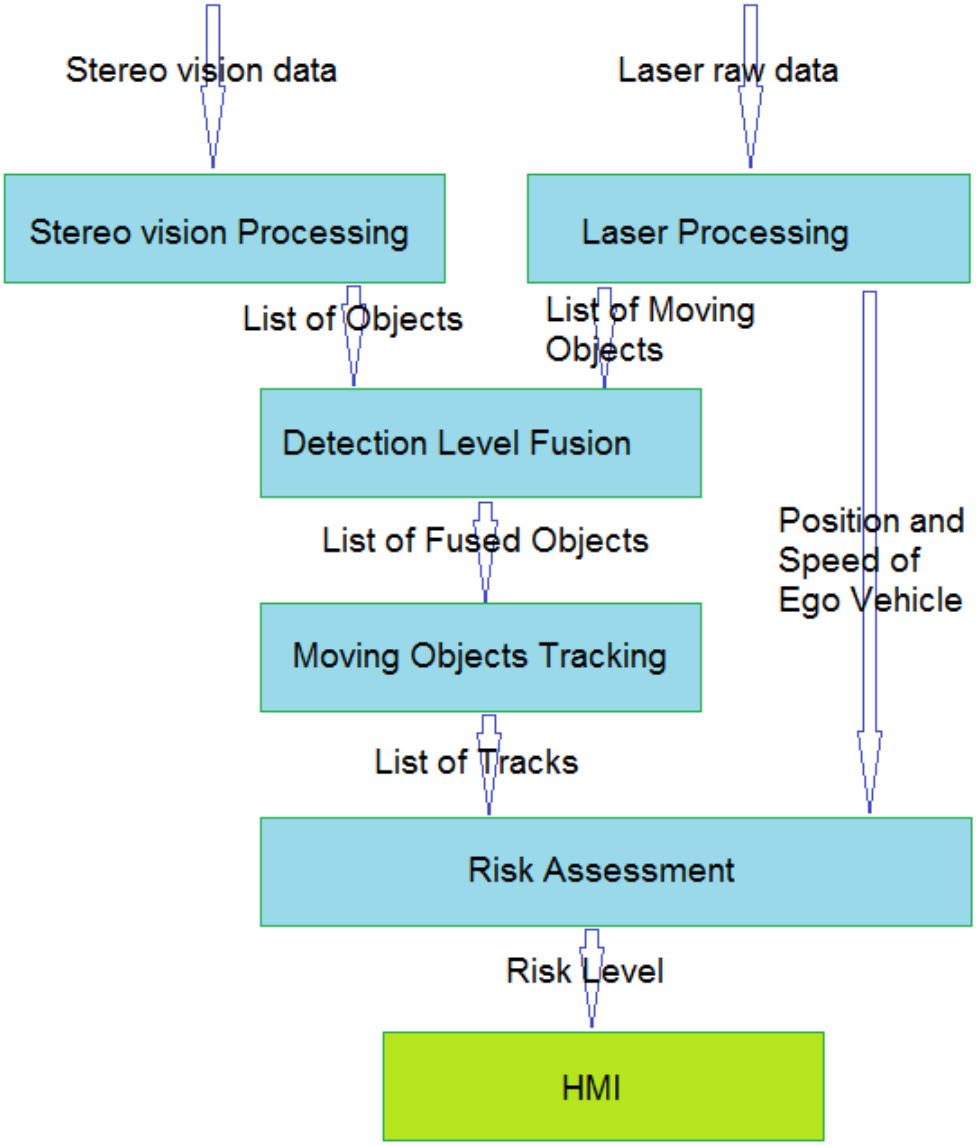

Figure 5.6: Architecture showing the dependence of different processing modules.

age acquisition process and provides, after the sensorial data processing, the primary information needed by the high level processing modules: 6D point information (3D position and 3D motion), ego motion estimation and intensity images at a rate of 20 frames per second. Using the rich output of the low-level architecture the two environment descriptions (structured and unstructured)are generated.

Obstacle detection

- 3D Points pre-processing

Using information from the digital elevation map the 3D points are classified according

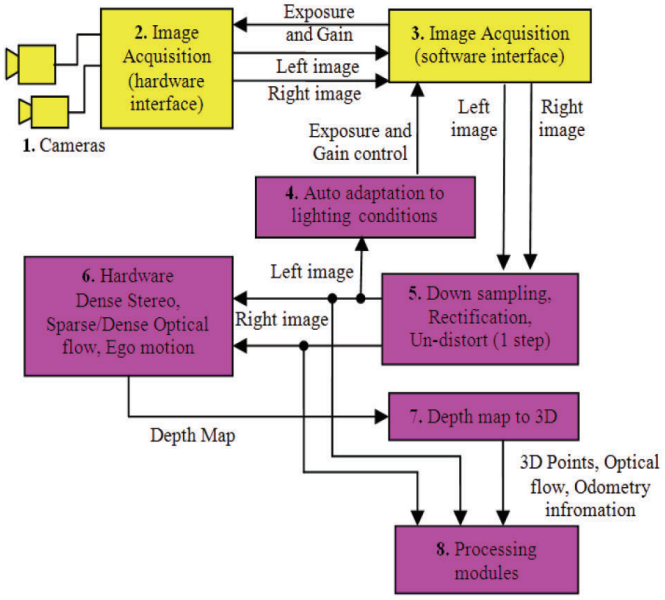

Figure 5.7: Low level architecture.

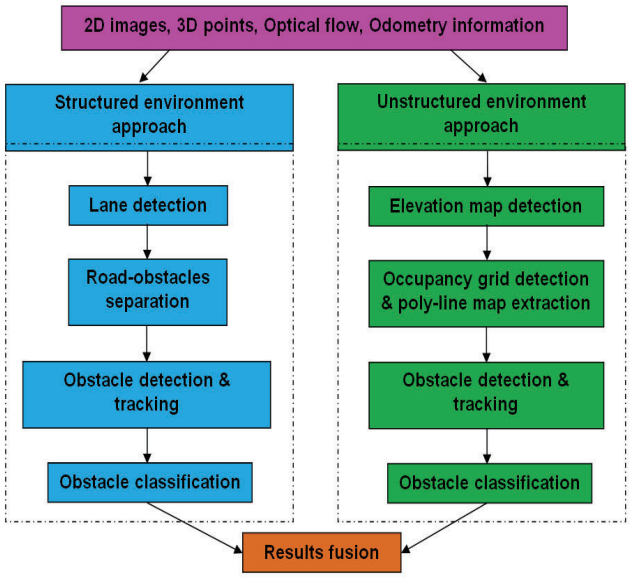

Figure 5.8: High level architecture.

to their position with regard to the detected road/navigable area plane. Optical flow provides motion information for a subset of these points (corner points)

- Obstacle detection

An improved obstacle detection technique was developed based on the fusion of 3D position information with 3D motion information. The obstacle detection algorithm extends the existing polar occupancy grid-based approach by augmenting it with motion

information. The occupied areas are fragmented into obstacles with cuboidal shape without concavities and only with 90° convexities.

- Relevant obstacles classification

The goal of obstacle classification is to recognize the relevant objects in an intersection. They have identified three classes of relevant objects: Pedestrian, Pole, Car. A generic classification system able to recognize in real-time each of these classes of objects has been developed (Nedevschi *et al.*, 2009a) (Figure 5.9).

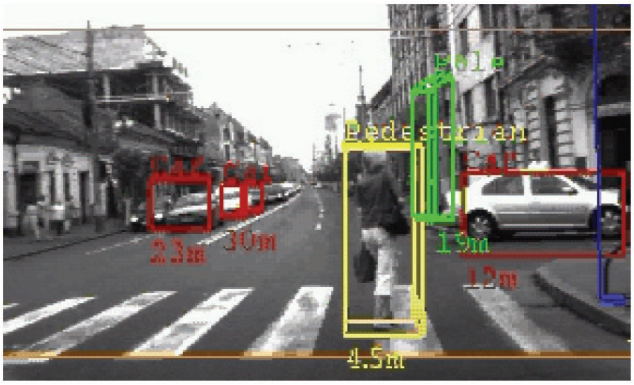

Figure 5.9: Output of classification: the predicted class.

- Obstacle representation

The obstacle are represented as cuboids carrying the following information:

- the cuboid's position, orientation and size,
- lateral and longitudinal speed,
- variance of the object center coordinates orientation and speed.
- tracking history (number of frames in which this object was previously seen)
- The detected obstacles are classified into: pedestrians, cars, poles and unknown

5.3.2 Lidar processing

For this project we have done a complete processing of lidar data to construct environment model, host vehicle localization and moving objects detection, the details of this processing were published in Baig *et al.* (2009, 2011) papers and are summarized below (detailed descriptions in chapter 3).

Environment mapping

As explained in 3^{rd} chapter, we have used an occupancy grid based environment mapping scheme for lidar data. In this scheme the environment of the robot is divided into rectangular cells and the mapping problem reduces to finding the occupancy state of each cell using lidar data and vehicle dynamics data accumulated over time. In this setting the occupancy state of each cell m is inferred knowing all the lidar measurements ($z_{1:t}$) and the trajectory of the vehicle ($x_{1:t}$) and is represented by the probability $P(m|x_{1:t}, z_{1:t})$. We further reduced this probability formulation into a recursive formula where new state of the cell depends on its previous state and value of inverse sensor model for current estimated position and current lidar data. This scheme called incremental mapping gives environment maps which are very good approximation of the actual environment.

Host vehicle localization

At each time step initial estimate of new vehicle position is made by integrating new values of translational and rotational velocities (v_t, ω_t) given by IMU sensor. However, to deal with drift errors we have used an incremental scan matching technique to correct vehicle pose in the built map. In this approach using a particle filter different samples of new vehicle position are selected from the vehicle motion model. From each of this new position current lidar scan is matched with the map built so far and a score function is calculated. The position giving best match (highest score) is taken as the corrected pose of the vehicle in the map.

Both mapping and localization are usually solved simultaneously and are known by the name of *SLAM* problem. The mapping and localization techniques summarized in these two subsections are detailed in 3^{rd} chapter and are collectively called maximum likelihood based *SLAM* solution.

Moving objects detection

Our moving objects detection scheme is based on the concept of finding inconsistencies between map built so far and new lidar data. To find these inconsistencies new lidar data is projected on the updated map from the current corrected position. The laser hits observed in free areas of the map are taken as belonging to moving objects. Then using a clustering technique these individual dynamic hits are clustered into objects. A list of such objects from each lidar scan are the detected moving objects.

5.3.3 Lidar and stereo vision fusion

Although we have implemented fusion between lidar and stereo vision sensors at three different levels, however only object detection level fusion (explained in section 3.3 of 3^{rd} chapter) was integrated on the VW demonstrator vehicle. In this scheme list of moving objects given by our lidar processing is fused with the list of objects given by stereo vision processing. The result of this fusion is a list of fused objects having more properties than their versions before fusion.

5.3.4 Moving objects tracking

As detailed in 3.3.4 section, the list of fused objects given by fusion module is input to the tracking module. Tracking module uses MHT based data association and IMM filtering techniques to track these moving objects.

5.3.5 Risk assessment

The risk assessment module provides an evaluation of the risk of a potential collision between the host vehicle and the objects that may be present in the driving environment. Our approach follows the work previously presented in (Ammoun and Nashashibi, 2009). This evaluation consists in the prediction of the environment for the future time instants and quantification of the risk associated to the detected collision situations: potential future collisions. It is considered that the driver has full control of the host vehicle and that the future driver behavior is unknown. The risk assessment is performed in the following sequential steps:

- scenario interpretation
- trajectory prediction
- collision detection
- risk quantification
- risk management

Scenario interpretation

The scenario interpretation consists of a representation of the current host vehicle state and local map of the surrounding environment composed of dynamic and static objects. This interpretation, that in most cases is incomplete and not very accurate, will

influence the performance of the risk assessment. The host vehicle state provides information about the position, heading, steering angle, velocity, acceleration, yaw rate of the host vehicle. The dynamic objects can be of 2 types: vehicles and pedestrians. The information about the static and dynamic objects is limited and it leads to some assumptions that will influence the trajectory prediction process:

- The objects of type vehicle keep current speed and direction: no information about steering angle, acceleration, yaw rate or blinkers is provided by the high level fusion or communications.
- The host vehicles and other dynamic objects trajectories are not constraint to follow the road: there is no information about the static objects like the road geometry and lanes description.

Trajectory prediction

Given the current scenario interpretation the host vehicle state and dynamic objects are modeled and integrated in time to provide a projection of the future environment representation. This integration time consists in predicting the trajectories of the dynamic objects, including host vehicle, just by using the current scenario interpretation. The future driver behavior is unknown and will not be predicted although it may affect the future trajectories. A trajectory of a dynamic object is a temporal sequence of object states for the future time instants. For each object a trajectory predicted from the current time t0 until a given time horizon t0+h where h is the total prediction time. The modeling of the trajectories is done taking into account the object type vehicle or pedestrian and associated information. The prediction of the vehicles trajectories, including the one of the host vehicle, is performed by using a bicycle dynamic model (Gillespie, 1992) integrated in time using the 4th order Runge-Kutta method that uses acceleration and steering rate commands as inputs. The initial vehicle state used to integrate the previous model is the one obtained at the time t0. Predict the movement of pedestrians is a substantially more difficult task (Nicolao *et al.*, 2007) than vehicles. Since a pedestrian can easily change direction no assumptions are made regarding the direction of its movement. The pedestrian is then modeled as a circle with a predefined radius centered at the initially detected pedestrian position at time t0, that will increase its radius in time proportionally to its initially estimated speed.

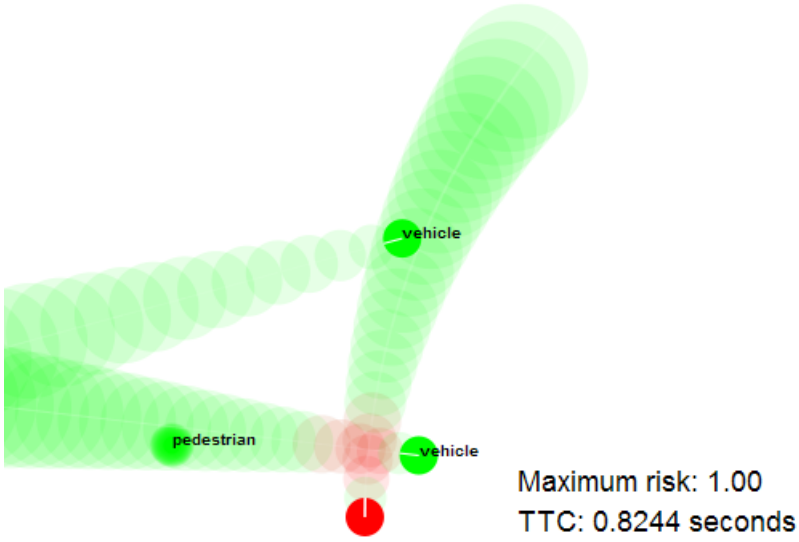

Figure 5.10: Example of a potential collision between the host vehicle (red circle on the bottom) and another vehicle (green circle on the bottom right).

Collision detection

The host vehicle and dynamic objects are represented as circles: the circle center is given by the object position, and the circle radius is set according with the object type, at a given moment in time. The position uncertainty of the objects is represented by an increase of the circles radius in function of the estimated traveled distance by the object. A potential collision is detected when the host vehicle circle intersects at least one circle of the dynamic objects at the same moment in time. Figure 5.10 gives an illustration of this process.

Risk quantification

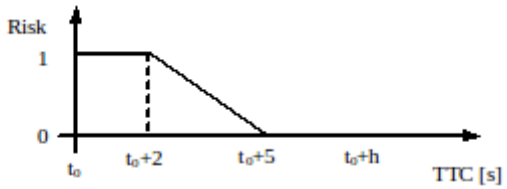

Figure 5.11: Relation between TTC and risk indicator.

The risk of collision is calculated for the nearest potential collision situation in time. For calculating this risk it is used the parameter time-to-collision (TTC) that corresponds to the duration between the current time t0 and the instant of when the first detected collision will occur. A consideration is taken that all objects keep their initial speeds

until the moment of the collision. The TTC is an important parameter because it can be compared to the driver and vehicle reaction times and provide a collision risk indicator. In our implementation the obtained TTC is compared with the commonly used total reaction time of 2 seconds: driver (1 sec) Zhang *et al.* (2006) and vehicle (1 sec). The risk estimation performed until a predefined time horizon t0+h and the risk indicator is given by the relation shown in figure 5.11.

Risk management

Based on quantification of the collision risk two strategies can be adopted to avoid or minimize the potential accident: Information or warning: advices are provided to the driver through the appropriate HMI visual, audio, or haptic feedback to avoid or reduce the risk of accident. Intervention: the automation takes momentarily control of the vehicle to perform an obstacle avoidance or collision mitigation manoeuvre. In our implantation only visual information is provided to the driver with periodical estimation of the collision risk for the given scenario interpretation.

5.4 System integration into demonstrator vehicle

Our final contribution in this project was the development of integration modules that could allow smooth communication between different components developed by different teams.

Volkswagen decided to use *Data Distribution Services (DDS)*[3] developed by *Real-Time Innovations (RTI)*. This communication layer application takes care of data transfer at low level and provides interfaces to send and receive data. In DDS, communication with higher level modules is implemented through *subscribers* and *publishers*. A *publisher* application allows a module to send its output on the DDS layer that will be used as input by some other module whereas a *subscribers* application allows to receive data from DDS layer published by some other module.

For this project the three teams implemented their components independently on different computers and these three computers were put in the demonstrator vehicle for integration. We implemented following subscriber and publisher modules to allow communication between these components implemented on three different computers in the demonstrator vehicle:

- *Lidar data subscriber:* a subscriber application to read raw laser data.

[3] `http://www.rti.com/products/dds/`

- *IMU subscriber:* was used to read inertial measurement unit data for host vehicle dynamics.
- *Stereo vision subscriber:* This module allowed us to read stereo vision objects published by Cluj team.
- *Fusion results publisher:* a publisher application that we implemented to transmit our fusion and map results on the network to be used by risk assessment module.
- *Risk assessment subscriber:* a subscriber module that we implemented for risk assessment module to read data published by our fusion module.

This integration scheme is shown in figure 5.12. Sensors shown in bottom layer publish their data on DDS layer for processing. Respective modules read this data through subscribers (violet arrows), process data and publish their results through publishers (blue arrows) on the DDS layer to be used by next module. These integration modules allowed a smooth communication between components developed by teams involved in the task.

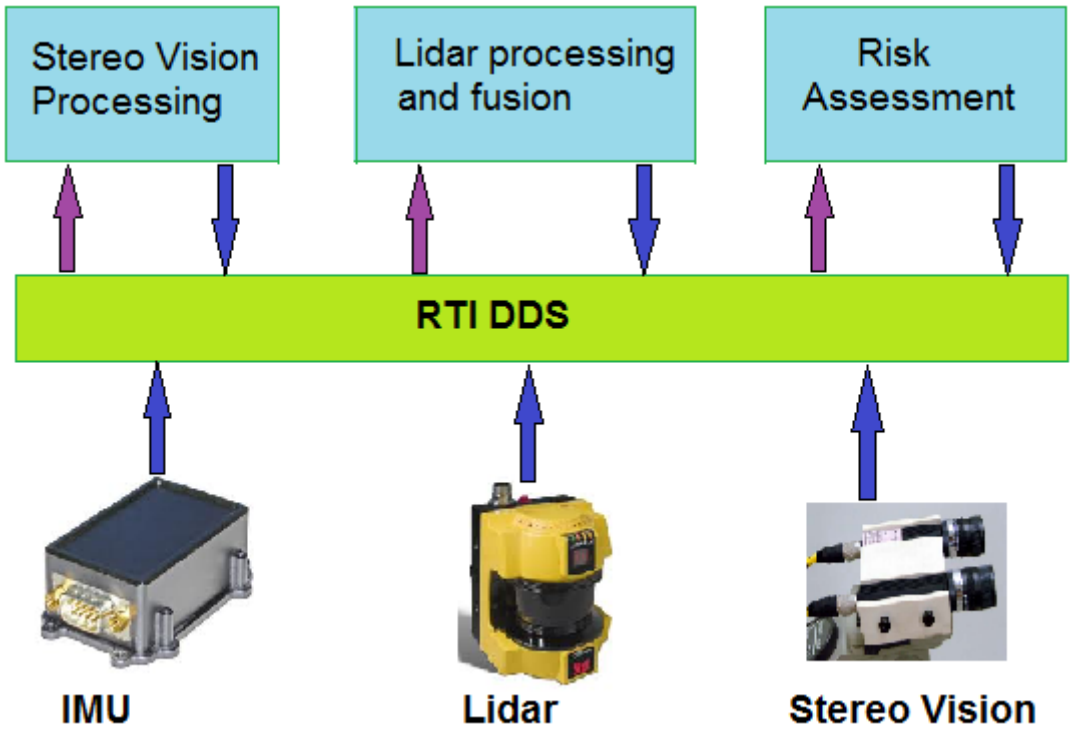

Figure 5.12: System Integration architecture showing communication between different modules through RTI DDS layer. Violet arrows are subscriber interfaces whereas blue arrows are publisher interfaces.

5.5 Synthesis

In this chapter we have briefly reviewed the application of our fusion work in the INTERSAFE-2 project. This is an *FP7* project funded by European Union and deals with safety at intersections. Intersections are still seen as a risky area in traffic networks and from 30 to 60% of all injury accidents and up to one third of fatalities occur at intersection. In this project two new concepts, other than equipped host vehicle, were also tested. These concepts include installing sensors on the intersections to observe areas not visible to the vehicle sensors and share these information with host vehicle to increase its perception range. The second concept was to share perception information between multiple equipped vehicles with the same objective of increasing each others perception range to avoid dangerous situations. This operation mode of sharing information is called cooperative mode. However if intersections are not equipped or there is no other equipped vehicle at the intersection to share information even then host vehicle should be able to operate in a self contained safety mode called independent mode. Different layers and modules for both of these modes are summarized in this chapter.

Our work in this project was related to perception layer of the vehicle's independent mode safety system. We performed complete lidar processing to build environment map and to find precise position of the host vehicle in this map to solve localization problem. We also developed techniques to find moving objects from laser data and perform fusion between lists of objects given by lidar processing and stereo vision processing done by University of Cluj. Finally the fused lists of objects were used to find tracks by solving tracking problem. The output of our perception module was used by a team working at INRIA Paris charged with the work of risk assessment to find dangerous situations and to take appropriate actions if need be. Finally we developed integration techniques using DDS technology by RTI to allow smooth communication between components developed by these partners. This project has been successfully completed and its final assessment took place on 17^{th} May 2011 in Wolfsburg Germany.

CHAPTER 6

Conclusion and Perspectives

6.1 Conclusion

In this thesis we have worked on the vehicle perception problem divided into *SLAM* and *DATMO* sub problems using lidar and vision (stereo and mono) sensors. Although some other people (like Vu (2009) and Wang (2004)) have also addressed these problems using single laser sensor however we have seen that the solution developed by them depend on the quality and type of the sensor used. They usually have used high resolution (<= 0.5°) laser scanner, however, these solutions do not give good results with low resolution (2°) and noisy laser scanner. We have reimplemented the solution developed by Vu (2009) to confirm that, although, *SLAM* results are acceptable but *DATMO* results have many false alarms and miss detections. The lidar scanner installed on the VW demonstrator used for INTERSAFE-2 project (on which I worked during last 3 years) other than having low resolution, has some other problems like: some times it reports reflections from white lane markings on the road giving false moving objects, it has relatively more longitudinal uncertainty giving irregular shape of even straight objects (like road curbs). Due to low resolution very few hits are observed on objects at distance making them appear as noise eventually causing miss detections. Since demonstrator used for INTERSAFE-2 project also has other sensors installed on it, especially stereo vision, so we proposed to solve problems of laser scanner by using data from both laser and stereo vision and by developing fusion techniques on different levels, primarily for this particular context.

In the setting of *SLAM* and *DATMO* based perception architecture we have noted that fusion can take place at three different levels. In chapter 2 after reviewing already existing solutions and techniques in the literature on this subject we give details of these three levels. They include: *Low level fusion* when only basic preprocessing of the sensors

data has been done. We propose to use occupancy grid based fusion technique at this level. *Object detection level fusion*, data of individual sensors are processed separately to detect objects and we propose to define a fusion function that will fuse these lists of objects to get fused list of objects. These fused objects have more properties than their previous versions giving good tracking results. *Track level fusion*, moving objects are detected and tracked for each sensor individually and fusion takes place on the lists of tracks. This fusion greatly helps to reduce the number of false tracks.

Chapter 3 gives details of these fusion levels for lidar and stereo vision sensors. For low level fusion we explain the inverse sensor model used for laser data to construct occupancy grid and give details of inverse sensor model that we have developed for stereo vision data that consists of 3D points, to build a compatible occupancy grid for stereo vision. We see that different fusion functions depending on the behavior of sensors can be used to combine these occupancy grids to get a fused occupancy grid. We also detail how this fused occupancy grid can be used as an input for *SLAM* module to update the map. For object level fusion we explain how to get moving objects from lidar data and define a fusion function for these objects and vision objects given by stereo processing done by an other team. We see that this list of fused objects has more properties and helps improve the tracking part. Finally track level fusion helps remove false tracks detected due to sensor problem in the critical common area of both lidar and stereo sensor.

In the same chapter we detail our fast road border detection technique to find road borders form laser data. The main advantage of detecting road borders is to remove false moving objects detected close to road borders due to longitudinal lidar uncertainty. We had noted that many false moving objects are detected close to road borders. The developed technique is very effective and can be used for any laser or similar sensors to find road borders and for constructing more intuitive environment model.

Chapter 4 gives details of fusion techniques for lidar and mono vision sensors. Although this work was not developed for INTERSAFE-2 data sets, instead we used a simulator for low level fusion and some imitated vision data for object level fusion, however we see that in particular context fusion between mono vision and lidar can be very useful. If the environment is structured and background can be separated from foreground then have suggest an inverse sensor model for mono vision to construct occupancy grid that can be fused with lidar occupancy grid. As explained in this chapter, occupancy grid constructed this way is especially useful if we need more belief for free space in-front of the vehicle. For object level fusion between laser and mono vision we suggest that images from mono camera can be used to get class information of

the detected moving objects in the laser and then a model based tracking can be done to get more smooth trajectories of moving objects. To speed up image processing we suggest to use transformations for laser to image to find regions of interests (ROI) corresponding to detected laser objects and only these ROIs should be used for the object classifier. We think that a track level fusion between laser and mono vision, although possible, however may not be interesting. As we cannot get easily and precisely depth information from single images so fusion at this level will only contribute classification information making it similar to object level fusion but at higher processing costs.

Chapter 5 reviews INTERSAFE-2 project, its system architecture and contribution of the partners who worked on Volkswagen demonstrator vehicle.

We see that In many of the works related to fusion, the fusion is done after tracking. The main advantage of this strategy is that fusion module can be designed independent of the sensors. This is very convenient to design generic methods to perform sensor data fusion. On the other hand, high level fusion has several drawbacks: to be able to perform fusion, we have to wait until an object is tracked which can take several frames. Another important problem is when an object is detected, it is sometimes not created for tracking due to numerous false alarms or miss detections. To overcome these problems we see that object level fusion is more appropriate. It is not too early (like low level fusion) where finding common representation scheme requires to develop similar processing models for even different sensors (for example finding an inverse sensor model for vision sensor that is compatible with that of lidar to construct compatible occupancy grids). And is not too late (like in case of track level fusion) that it may take many frames to confirm an object as a moving target, especially in the case when a moving object goes out of view before it is confirmed as a valid dynamic object. Other information inferred from data (like road borders from laser data) can help improve the quality of the perception results.

In this thesis we have presented a fusion based solution for *SLAM* and *DATMO* based perception architecture for autonomous vehicles moving fast in urban environments for particular sensor constraints. Although there exist other fusion algorithms however as rightly pointed out by Hall and Garga (1999) *there is no such thing as a magic or golden data fusion algorithm*, a need of new algorithms will keep on arising for different sensors and/or target environments. We hope that our work in this thesis has explored another dimension in fusion and can be useful for many similar situations as they are not rare in robotics field.

6.2 Perspectives

Work done in this thesis can be extended in many different ways. Firstly we see that many of the existing fusion techniques are based on Bayesian filters. We see that on both object and track level these filters can be used to remove many of the false objects before fusion resulting in better results. Although this will incur an overhead of extra processing but it may be worth it in certain situations like when data is very much noisy.

Secondly we will like to extend our road border detection technique by adding tracking them over time to get smooth borders even if data is missing. And to detect real moving objects detected on or beyond road borders like pedestrians on footpath.

Although Bayesian approach used in this thesis is fast, however, some times it does not handle the conflicting information in a more useful way, for example if there is conflict in two occupancy grids where one sensor sees an object whereas second sensor sees this place empty. Then a Bayesian combination of the grids will give the occupancy state of these cells between the two pre fusion versions, making these information of little use. We solved this kind of situations by using knowledge of sensor properties and defining custom rules. However Dempster-Shafer and evidence theory based on belief functions handle such situations in a more systematic way. Another perspective is to identify situation where these approaches can give better results than the Bayesian approach and use them to solve data fusion problem.

Stereo vision system used for this work is also capable of giving lane and other road marking (arrows, stop lines) information, however, for current work we have ignored such information. Moreover occupancy grid based map representation technique used for environment representation is not capable of using these information because it can only be used to see if a place is occupied or empty. We also plan to use new map representation techniques capable of making use of such useful information and hence giving more realistic maps.

References

Abuhadrous I., Nashashibi F., Laurgeau C., and Chinchole M. Multi-sensor data fusion for land vehicle localization using rtmaps. In *IEEE Intelligent Vehicles Symposium Proceedings.*, pages 339 – 344, June 2003.

Ammoun Samer and Nashashibi Fawzi. Real time trajectory prediction for collsion risk estimation between vehicles. In *IEEE International Conference on Intelligent Computer Communication and Processing*, 2009.

Ayache N. and Faugeras O.D. Building, registrating, and fusing noisy visual maps. *Int. J. Rob. Res.*, 7:45–65, December 1988. ISSN 0278-3649. doi: 10.1177/027836498800700605. URL `http://dl.acm.org/citation.cfm?id=55067.55071`.

Aycard O., Baig Q., Bota S., Nashashibi F., Nedevschi S., Pantilie C., Parent M., Resende P., and Vu Trung-Dung. Intersection safety using lidar and stereo vision sensors. In *Intelligent Vehicles Symposium (IV), 2011 IEEE*, pages 863 –869, june 2011. doi: 10.1109/IVS.2011.5940518.

Baig Q. and Aycard O. Low level data fusion of laser and monocular color camera using occupancy grid framework. In *International Conference on Control, Automation, Robotics and Vision (ICARCV)*, Singapore, December 2010.

Baig Q., Vu TD., and Aycard O. Online localization and mapping with moving objects detection in dynamic outdoor environments. In *IEEE Intelligent Computer Communication and Processing (ICCP)*, Cluj-Napoca, Romania, August 2009.

Baig Q., Aycard O., Vu Trung Dung, and Fraichard Thierry. Fusion between laser and stereo vision data for moving objects tracking in intersection like scenario. In *International Conference on Intelligent Vehicles (IV)*, Baden-Baden, Germany, June 2011.

Baltzakis H., Argyros A., and Trahanias P. Fusion of laser and visual data for robot motion planning and collision avoidance. *Mach. Vision Appl.*, 15(2):92–100, 2003. ISSN 0932-8092. doi: http://dx.doi.org/10.1007/s00138-003-0133-2.

Bar-Shalom Y. and Chen Huimin. Multisensor track-to-track association for tracks with dependent errors. In *43rd IEEE Conference on Decision and Control. CDC.*, volume 3, pages 2674 – 2679 Vol.3, dec. 2004. doi: 10.1109/CDC.2004.1428864.

Bar-Shalom Y. and Li X. *Multitarget Multisensor Tracking : Principles and Techniques*. YBS Publishing, 1995.

Bar-Shalom Y. and Tse E. Tracking in a cluttered environment with probabilistic data association. In *Proceedings of the 4th Symposium on nonlinear estimation theory and its applications*, 1974.

Bar-Shalom Y. and Tse E. Tracking in a cluttered environment with probabilistic data association. *Automatica*, 11, 1975.

Bar-Shalom Y., Daum F., and Huang J. The probabilistic data association filter. *Control Systems, IEEE*, 29(6):82 –100, December 2009.

Blackman S. and Popoli R. *Design and Analysis of Modern Tracking Systems*. Artech House, 1999.

Blackman S. S. Multiple hypothesis tracking for multiple target tracking. *Aerospace and Electronic Systems Magazine, IEEE*, 19(1):5–18, 2004.

Blair W.D. and Kazakos D. Tracking maneuvering targets with multiple, intermittent sensors. In *Signals, Systems and Computers, 1993. 1993 Conference Record of The Twenty-Seventh Asilomar Conference on*, volume 1, pages 258 –262, nov 1993.

Blanc C., Trassoudaine L., Guilloux Y. Le, and Moreira R. Track to track fusion method applied to road obstacle detection. In *IEEE International Conference on Information Fusion*, Stockholm, Sweden, July 2004.

Braillon Christophe, Usher Kane, Pradalier CÃľdric, Crowley James L., and Laugier Christian. Fusion of stereo and optical flow data using occupancy grids. In *In Proceedings of the IEEE Intelligent Transportation Systems (ITSC), Toronto (CA*, 2006.

Brooks R. Visual map making for a mobile robot. In *IEEE International Conference on Robotics and Automation*, volume 2, pages 824 – 829, March 1985. doi: 10.1109/ROBOT.1985.1087348.

Burlet J., Vu TD., and Aycard O. Grid-based localization and online mapping with moving objects detection and tracking. Technical report, INRIA-UJF, 2007.

Chanier Francois, Checchin Paul, Blanc Christophe, and Trassoudaine Laurent. Map fusion based on a multi-map slam framework. *2008 IEEE International Conference on Multisensor Fusion and Integration for Intelligent Systems*, pages 533–538, 2008. URL `http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=4648050`.

Cole D.M. and Newman P.M. Using laser range data for 3d slam in outdoor environments. In *ICRA 2006. Proceedings 2006 IEEE International Conference on Robotics and Automation, 2006.*, pages 1556 –1563, may 2006. doi: 10.1109/ROBOT.2006.1641929.

Cox Ingemar J. A review of statistical data association techniques for motion correspondence. *International Journal of Computer Vision*, 10:53–66, 1993.

Cox Ingemar J. and Hingorani Sunita L. An efficient implementation and evaluation of reid's multiple hypothesis tracking algorithm for visual tracking. *Pattern Recognition, 1994. Vol. 1*, 1, 1996.

Crowley J. L., Stelmaszyk P., and Discours C. Measuring image flow by tracking edge-lines. In *International Conference on Computer Vision*, 1988.

Csorba Michael. *Simultaneous Localisation and Map Building*. PhD thesis, Balliol College, University of Oxford, Oxbord, United Kindom, 1997.

Danescu R., Oniga F., Nedevschi S., and Meinecke M.-M. Tracking multiple objects using particle filters and digital elevation maps. In *Intelligent Vehicles Symposium, 2009 IEEE*, pages 88 –93, june 2009.

Deriche Rachid and Faugeras Olivier. Tracking line segments. *Image Vision Comput.*, 8: 261–270, January 1991.

Deselaers Thomas, Heigold Georg, and Ney Hermann. Object classification by fusing svms and gaussian mixtures. *Pattern Recogn.*, 43:2476–2484, July 2010. ISSN 0031-3203.

Deveza R., Thiel D., Russel R.A., and Mackay-Sim A. Odour sensing for robot guidance. *International Journal of Robotics Research*, 13(3):232–239, 1994.

Dissanayake M. W. M. Gamini, Newman Paul, Clark Steven, Durrant-whyte Hugh F., and Csorba M. A solution to the simultaneous localization and map building (slam) problem. *IEEE Transactions on Robotics and Automation*, 17:229–241, 2001.

Durrant-Whyte H. Integrating distributed sensor information, an application to a robot system coordinator. In *IEEE International Conference on Systems, Man and Cybernetics.*, pages 415 – 419, 1985.

Durrant-Whyte H. Consistent integration and propagation of disparate sensor observations. In *IEEE International Conference on Robotics and Automation*, volume 3, pages 1464 – 1469, April 1986.

Durrant-Whyte H. F. Uncertain geometry in robotics. *IEEE Journal on Robotics and Automation*, 4:23–31, 1988.

Durrant-Whyte H. F., Rao B. S. Y., and Hu H. Toward a fully decentralized architecture for multi-sensor data fusion. In *IEEE Int. Conf. Robot. Automat.*, pages 1331–1336, Cincinnati, OH, USA, 1990.

Durrant-Whyte H.F. An autonomous guided vehicle for cargo handling applications. *International Journal of Robotics Research*, 15(5):407–440, 1996.

Durrant-Whyte Hugh and Bailey Tim. Simultaneous localisation and mapping (slam): Part i the essential algorithms. *History*, 13(2):1–9, 2006. URL `http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.128.4195&rep=rep1&type=pdf`.

Elfes A. *Occupancy grids: a probabilistic framework for robot percpetion and navigation*. PhD thesis, Carnegie Mellon University, 1989a.

Elfes A. Using occupancy grids for mobile robot perception and navigation. *IEEE Computer, Special Issue on Autonomous Intelligent Machines*, pages 46–57, June 1989b.

Evers H. and Kasties G. Differential gps in a real-time land vehicle environment - satellite based van carrier location system. *IEEE Aerospace and Electrical Systems Magazine*, 9(8):26–32, 1994.

Fayad Fadi and Cherfaoui VÃl'ronique. Object-level fusion and confidence management in a multi-sensor pedestrian tracking system. In Hahn Hernsoo, Ko Hanseok, and Lee Sukhan, editors, *Multisensor Fusion and Integration for Intelligent Systems*, volume 35 of *Lecture Notes in Electrical Engineering*, pages 15–31. Springer Berlin Heidelberg, 2009. ISBN 978-3-540-89859-7.

Filipovych Roman and Ribeiro Eraldo. Probabilistic combination of visual cues for object classification. In *Proceedings of the 3rd international conference on Advances in visual computing - Volume Part I*, ISVC'07, pages 662–671, Berlin, Heidelberg, 2007. Springer-Verlag. ISBN 3-540-76857-2, 978-3-540-76857-9.

Floudas N., Lytrivis P., Polychronopoulos A., and Amditis A. On the track-to-track association problem in road environments. In *10th International Conference on Information Fusion*, pages 1–8, july 2007.

Fortmann T. E. and Baron S. Problems in multi-target sonar tracking. In *IEEE Conference on Decision and Control*, pages 1566–1573, January 1979.

Fuerstenberg Kay Ch., Dietmayer Klaus C. J., Eisenlauer Stephan, and Willhoeft Volker. Multilayer laserscanner for robust object tracking and classification in urban. In *9th World Congress on Intelligent Transport Systems, ITS*, pages 7–8, Chicago, USA, 2002.

Gérossier Franck, Checchin Paul, Blanc Christophe, Chapuis Roland, and Trassoudaine Laurent. Trajectory-oriented ekf-slam using the fourier-mellin transform applied to microwave radar images. In *Proceedings of the 2009 IEEE/RSJ international conference on Intelligent robots and systems*, IROS'09, pages 4925–4930, Piscataway, NJ, USA, 2009. IEEE Press. ISBN 978-1-4244-3803-7. URL `http://dl.acm.org/citation.cfm?id=1732643.1732850`.

Gidel S., Blanc C., Chateau T., Checchin P., and Trassoudaine L. Non-parametric laser and video data fusion: Application to pedestrian detection in urban environment. In *12th International Conference on Information Fusion*, pages 626 –632, july 2009.

Gillespie T.D. *Fundamentals of Vehicle Dynamics, Society of Automotive Engineers*. 1992.

Gordon N.J., Salmond D.J., and Smith A.F.M. Novel approach to nonlinear/non-gaussian bayesian state estimation. *Radar and Signal Processing, IEE Proceedings F*, 140(2):107 –113, apr 1993. ISSN 0956-375X.

Gros X. E. *NDT Data Fusion*. Arnold, 1996.

Gupte S., Masoud O., Martin R.F.K., and Papanikolopoulos N.P. Detection and classification of vehicles. *Intelligent Transportation Systems, IEEE Transactions on*, 3(1):37 –47, mar 2002. ISSN 1524-9050. doi: 10.1109/6979.994794.

Hall D. L. and Garga A. K. Pitfalls in Data Fusion (and How to Avoid Them). In *Proceedings of the 2nd International Conference on Information Fusion âĂŞ FUSION'99*, volume 1, pages 429–436, 1999.

Hall D. L. and Llinas J. An introduction to multisensor data fusion. *Proceedings of the IEEE*, 85(1):6–23, 1997.

Hall David L. and Llinas James. *Handbook of Multisensor Data Fusion*. CRC Press, June 2001. ISBN 0849323797.

Hashimoto Minoru and Paul Richard. Integration of multi-sensor manipulator actuator information for robust robot control systems. In *Annual Conference of Japan Robotics Society*, pages 393 – 396, 1987.

Hebert P., Betge-Brezet S., and Chatila R. Decoupling odometry and exteroceptive perception in building a global map of a mobile robot: The use of local maps. In *IEEE International Conference on Robotics and Automation*, Minneapolis, Minnesota, USA, April 1996.

Jain Ramesh, Martin W. N., and Aggarwal J. K. Segmentation through the detection of changes due to motion. *Computer Graphics and Image Processing*, 11(1):13–34, September 1979.

Kalman R.E. A new approach to linear filtering and prediction problems. *Journal of basic Engineering*, 35, March 1960.

Kirchner A. and Heinrich Th. Model based detection of road boundaries with a laser scanner. 1998.

Kleeman L. Optimal estimation of position and heading for mobile robots using ultrasonic beacons and dead reckoning. In *IEEE International Conference on Robotics and Automation*, Nice, France, May 1992.

Konolige Kurt. Improved occupancy grids for map building. *Auton. Robots*, 4(4):351–367, 1997.

Konstantinova Pavlina, Udvarev Alexander, and Semerdjiev Tzvetan. A study of a target tracking algorithm using global nearest neighbor approach. In *Proceedings of the 4th international conference conference on Computer systems and technologies: e-Learning*, pages 290–295, New York, NY, USA, 2003. ACM. URL `http://dl.acm.org/citation.cfm?id=973620.973668`.

Kriegman D.J., Triendl E., and Binford T.O. A mobile robot: Sensing, planning and locomotion. In Cox I.J. and Wilfon G.T., editors, *Autonomous Robot Vehicles*, pages 450–458. Springer-Verlag, 1990.

Labayrade Raphaël, Royere Cyril, Gruyer Dominique, and Aubert Didier. Cooperative fusion for multi-obstacles detection with use of stereovision and laser scanner. *Auton. Robots*, 19:117–140, September 2005. ISSN 0929-5593. doi: 10.1007/s10514-005-0611-7. URL `http://dl.acm.org/citation.cfm?id=1073557.1073563`.

Leonard J.J. and Durrant-Whyte H.F. *Directed Sonar Sensing for Mobile Robot Navigation*. Kluwer Academic Publishers, 1992.

Li Dalong. Moving objects detection by block comparison. In *Electronics, Circuits and Systems, 2000. ICECS 2000. The 7th IEEE International Conference on*, volume 1, pages 341–344, 2000.

Luo Ren, Lin Min-Hsiung, and Scherp R. The issues and approaches of a robot multi-sensor integration. In *IEEE International Conference on Robotics and Automation*, volume 4, pages 1941 – 1946, March 1987.

Mitchell H B. *Multi-Sensor Data Fusion: An Introduction*. Springer, 2007. ISBN 9783540714637.

Mittal A. and Davis L.S. M_2 Tracker: a multi-view approach to segmenting and tracking people in a cluttered scene. *International Journal of Computer Vision*, 51(3), 2003.

Montemerlo Michael and Thrun Sebastian. Fastslam: A factored solution to the simultaneous localization and mapping problem. In *In Proceedings of the AAAI National Conference on Artificial Intelligence*, pages 593–598. AAAI, 2002.

Moras J., Cherfaoui V., and Bonnifait P. Credibilist occupancy grids for vehicle perception in dynamic environments. In *2011 IEEE International Conference on Robotics and Automation, Shanghai International Conference Center*, pages 84–89, Shanghai, Chine, May 9-13 2011a. URL `http://hal.archives-ouvertes.fr/hal-00615303/en/`.

Moras J., Cherfaoui V., and Bonnifait P. Moving objects detection by conflict analysis in evidential grids. In *2011 Intelligent Vehicles Symposium*, pages 1120–1125, Baden-Baden, Allemagne, June 5-9 2011b. URL `http://hal.archives-ouvertes.fr/hal-00615304/en/`.

Moravec Hans. *Obstacle Avoidance and Navigation in the Real World by a Seeing Robot Rover*. Phd thesis, Computer Science Department Stanford University, California, USA, March 1980.

Nakamura Y. and Zu Y. Geometrical fusion method for multi-sensor robotic systems. In *IEEE International Conference on Robotics and Automation*, volume 2, pages 668 –673, May 1989. doi: 10.1109/ROBOT.1989.100061.

Nedevschi S., Danescu R., Marita T., Oniga F., and Bota S. Stereovision-based sensor for intersection assistance. In *Advanced Microsystems for Automotive Applications 2009*, pages 129–163. Springer Berlin Heidelberg, 2009a.

Nedevschi S., Marita T., Danescu R., Oniga F., and Bota S. On-board stereo sensor for intersection driving assistance. architecture and specification. In *IEEE Intelligent Computer Communication and Processing (ICCP)*, pages 409–416, Cluj-Napoca, Romania, August 2009b.

Nicolao G. De, Ferrara A., and Giacomini L. Onboard sensor-based collision risk assessment to improve pedestriansâĂŹ safety. *IEEE Transactions on Vehicular Technology*, 56 (5):2405–2413, 2007.

Nilsson Nils J. Shakey the robot. Technical report, SRI - Artificial Intellgence Center, Computer Science and Technology Division, April 1984.

Petrovskaya Anna and Thrun Sebastian. Model based vehicle detection and tracking for autonomous urban driving. *Autonomous Robots*, 26(2):123–139, 2009.

Pinto Nicolas, Cox David D, and DiCarlo James J. Why is real-world visual object recognition hard? *PLoS Computational Biology*, 4(1):6, 2008.

Poppen R.F. Using partial or nonstandard gps solutions in automotive navigation. In *International Technical Meeting of the Satellite Division of the Institute of Navigation*, Salt Lake City, UT, USA, September 1993.

Reid D. An algorithm for tracking multiple targets. *Automatic Control, IEEE Transactions on*, 24(6):843–854, 1979. URL `http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=1102177`.

Reid D. B. A multiple hypothesis filter for tracking multiple targets in a cluttered environment. Technical Report D-560254, Lockheed Missiles and Space Company Report, 1977.

Rencken W.D. Concurrent localisation and map building for mobile robots using ultrasonic sensors. In *IEEE Conference on Intelligent Robots and Systems*, Yokohama, Japan, July 1993.

Russel R.A. Mobile robot guidance using a short lived heat trail. *Robotica*, 11(5):427–431, 1993.

Russel R.A., Thiel D., and Mackay-Sim A. Sensing odour trails for mobile robot navigation. In *IEEE International Conference on Robotics and Automation*, San Diego,CA, USA, May 1994.

S.Blackman and Popoli R. *Design and Analysis of Modern Tracking Systems*. A K Peters, 1999.

Shooter C. and Reeve J. Intersafe-2 architecture and specification. In *Intelligent Computer Communication and Processing, 2009. ICCP 2009. IEEE 5th International Conference on*, pages 379 –386, aug. 2009. doi: 10.1109/ICCP.2009.5284730.

Singer R., Sea R., and Housewright K. Derivation and evaluation of improved tracking filter for use in dense multitarget environments. *Information Theory, IEEE Transactions on*, 20(4):423 – 432, July 1974. ISSN 0018-9448. doi: 10.1109/TIT.1974.1055256.

Singer Robert A. and Stein John J. An optimal tracking filter for processing sensor data of imprecisely determined origin in surveillance systems. In *Decision and Control, 1971 IEEE Conference on*, volume 10, pages 171 –175, Miami Beach, FL, USA, December 1971.

Smith R. and Cheeseman P. On the representation of spatial uncertainty. *International Journal of Robotic Research*, 5(4):56–68, 1987.

Smith R., Self M., and Cheeseman P. *Estimating uncertain spatial relationships in robotics*, pages 167–193. Springer-Verlag New York, Inc., New York, NY, USA, 1990. ISBN 0-387-97240-4.

Taleghani Sanaz, Aslani Siavash, and Shiry Saeed. *Robust Moving Object Detection from a Moving Video Camera Using Neural Network and Kalman Filter*, pages 638–648. Springer-Verlag, Berlin, Heidelberg, 2009. ISBN 978-3-642-02920-2. doi: 10.1007/978-3-642-02921-9_55. URL `http://dl.acm.org/citation.cfm?id=1575210.1575268`.

Thrun S., Burgard W., and Fox D. *Probabilistic Robotics (Intelligent Robotics and Autonomous Agents)*. The MIT Press, September 2005. ISBN 0262201623. URL `http://www.amazon.co.uk/exec/obidos/ASIN/0262201623/citeulike-21`.

Tsumura T. Survey of autonomous guided vehicles in japanese factory. In *IEEE International Conference on Robotics and Automation*, San Fransisco,CA, USA, April 1986.

Vandorpe J., Xu H., van Brussel H., and Aertbelien E. Positioning of the mobile robot lias using natural landmarks and a 2d rangefinder. In *IEEE International Conference on Multisensor Fusion and Integration for Intelligent Systems*, Washington, DC, USA, December 1996.

Vu TD., Aycard O., and Appenrodt N. Online localization and mapping with moving objects tracking in dynamic outdoor environments. In *IEEE International Conference on Intelligent Vehicles*, 2007.

Vu TD., Burlet J., and Aycard O. Mapping of environment, detection and tracking of moving objects using occupancy grids. In *Workshop on Intelligent Transport*, 2008.

Vu TD., Burlet J., and Aycard O. Grid-based localization and local mapping with moving objects detection and tracking. *International Journal on Information Fusion, Elsevier*, 2009.

Vu Trung-Dung. *Vehicle Perception: Localization, Mapping with Detection, Classification and Tracking of Moving Objects*. PhD thesis, Institut National Polytechnique de Grenoble, Grenoble, France, September 2009.

Vu Trung-Dung and Aycard Olivier. Laser-based detection and tracking moving object using data-driven markov chain monte carlo. In *IEEE International Conference on Robotics and Automation (ICRA)*, Kobe, Japan, May 2009.

Wang C.-C. *Simultaneous Localization, Mapping and Moving Object Tracking*. PhD thesis, Robotics Institute, Carnegie Mellon University, Pittsburgh, PA, April 2004.

Wang C.-C., Thorpe C., and Thrun S. Online simultaneous localization and mapping with detection and tracking of moving objects: Theory and results from a ground vehicle in crowded urban areas. volume 1, 2003.

Weiss T., Schiele B., and Dietmayer K. Robust driving path detection in urban and highway scenarios using a laser scanner and online occupancy grids. In *Intelligent Vehicles Symposium (IV)*, Istanbul, Turkey, June 2007.

Wijesoma W.S., Kodagoda K.R.S., Balasuriya A.P., and Teoh E.K. Road edge and lane boundary detection using laser and vision. In *International Conference on Intelligent Robots and Systems*, Hawaii, USA, November 2001.

Yguel M., Aycard O., Raulo D., and Laugier C. Grid based fusion of offboard cameras. In *IEEE International Conference on Intelligent Vehicules*, 2006.

Yi Zou, Khing Ho Yeong, Seng Chua Chin, and Wei Zhou Xiao. Multi-ultrasonic sensor fusion for mobile robots. In *Proceedings of the IEEE Intelligent Vehicles Symposium*, pages 387 –391, 2000a.

Yi Zou, Khing Ho Yeong, Seng Chua Chin, and Wei Zhou Xiao. Multi-ultrasonic sensor fusion for autonomous mobile robots, sensor fusion: Architectures, algorithms and applications. In *IEEE Intelligent Vehicles Symposium 2000*, pages 387–391, 2000b.

Zhang Y., Antonsson E. K., and Grote K. A new threat assessment measure for collision avoidance systems. In *IEEE International Intelligent Transportation Systems Conference*, 2006.

Zhang Z. Iterative point matching for registration of free-form curves and surfaces. *International Journal of Computer Vision*, 13(2):119–152, 1994.