

Final Project Report

Perception System for Autonomous Driving

(5/6/2017)

Team A (Team Aware)

Amit Agarwal, Harry Golash, Yihao Qian, Menghan Zhang, Zihao Zhang

Masters of Science in Robotic Systems Development

Carnegie Mellon University

(Sponsored by Delphi Automotive)



Abstract

This report describes and summarizes the progress made by our team (Team A) while working on our Master of Science in Robotics Systems Development capstone project at Carnegie Mellon University - “Perception System for Autonomous Driving”. We developed and tested a standalone perception system for autonomous cars using a pair of cameras and a radar sensor. Through the fusion of stereo vision and radar sensors, our system identifies objects in most conditions, while still being relatively inexpensive, compact, and efficient.

We use a pair of identical PointGrey cameras and a Delphi radar unit. Vision-based object detection and classification is done using the SSD algorithm, and the stereo-vision depth-map is calculated using the SGBM algorithm. Radar detection-level data are processed through a Kalman filter based on the constant velocity assumption. Sensor fusion is then achieved with position-based Multiple Hypothesis Tracking (MHT). We use rviz in ROS to display the live camera feed and a grid showing the identified objects by colored markers. This GUI updates information at 20 Hz.

Our unified system simultaneously detects near- and far-range objects. Stereo vision reliably detects and classifies vehicles and pedestrians ($> 70\%$ accuracy) up to ~ 30 meters away, but it gives unreliable depth estimates ($< 30\%$ error). Meanwhile, radar depth and velocity calculations are very accurate ($< 5\%$ and $< 15\%$ error, respectively) up to 120 meters, but the radar may ignore pedestrians occasionally. We thus use position-based MHT for sensor fusion, and so improve our total vehicle-detection accuracy by over 10%.

Table of contents

1. Project description	00
2. Use case	00
3. System requirements	00
3.1. Subsystem requirements	00
3.2. Changes made	00
3.3. Performance requirements	00
3.4. Non-functional requirements	00
4. Functional Architecture	00
4.1. Data processing	00
4.2. Perception	00
5. System-level trade studies	00
5.1. Vision subsystem	00
5.2. Radar subsystem	00
5.3. Computing subsystem	00
6. Cyber-physical Architecture	00
6.1. Hardware	00
6.2. Software	00
7. System description and evaluation	00
7.1. Subsystem descriptions	00
7.1.1. Vision subsystem	00
7.1.2. Radar subsystem	00
7.1.3. Computing subsystem	00
7.2. Modeling, analysis, and testing	00
7.3. SVE performance evaluation	00
7.4. Strong and weak points	00
8. Project Management	00
8.1. Schedule	00
8.2. Budget	00
8.3. Risk Management	00
9. Conclusions	00
9.1. Lessons learned	00
9.2. Future work	00
10. References	00

1. Project description

1.1 Background Information

Current self-driving cars such as those used by Google and Uber have many limitations in their perception systems. As can be seen in Fig.1. below, existing sensor racks are bulky, expensive, and hard to maintain. This is due to the large number of redundant sensors used by such systems in order to avoid misreading the environment under various driving conditions.

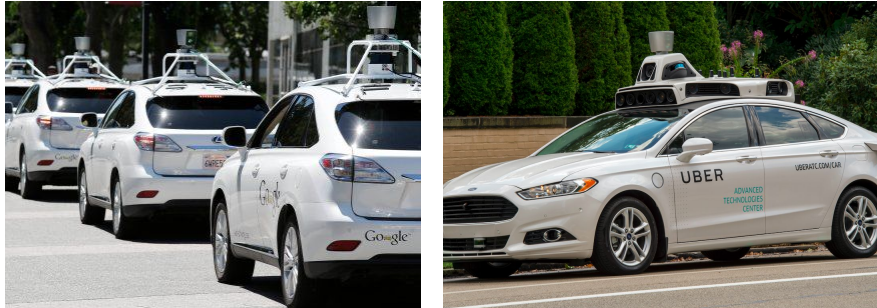


Fig. 1. Current sensor racks used by Google (left) and Uber (right) autonomous cars

Another issue with existing automotive perception systems is that these systems do not adequately apply sensor fusion of complementary sensor systems. Consequently, even an advanced autonomous vehicle can fail if all its sensors are blinded by a single stimulus. For example, consider Tesla's self-driving car crash that occurred when the test car's LIDAR and vision sensors were overpowered and compromised by sunlight reflected from a crossing truck.

In comparison, Fig. 2. below shows the self-driving car developed by our sponsors, Delphi Automotive. This was one of the first autonomous vehicles to drive cross-country across the US. In this case, the sensors have been installed on the vehicle while preserving its form and aesthetics. This was possible by using fewer sensors (meaning less redundancy) through smarter programming and sensor fusion, which all makers of autonomous vehicles seek to achieve.



Fig. 2. Delphi's self-driving SUV has integrated sensors

1.2 Project Information

In the last section, we identified the **user needs** of autonomous vehicle developers to have an inexpensive, reliable, and minimalistic automotive perception system that is easy to integrate and test. Developers desire a system with intelligent sensor fusion both to reduce the number of total sensors used and to minimize the risk of misreading the environment. In this project, we combine the input from multiple sensors to create an improved perception system that can be installed in any car for autonomous driving purposes. Our minimalistic system uses fewer sensors, making it less expensive and easier to integrate compared existing solutions.

We know that stereo vision and radar are typically used for short-range and long-range perception, respectively. The **result** of our project is therefore a standalone perception system that combines these two sensor systems to create a system that can simultaneously perceive in both the long and short range. Using rviz in ROS, the user can view the detected vehicles and pedestrians in the driving environment along with their positions and velocities in real-time. We found that the radar and the vision subsystems complement each other well for object detection. The vision system identifies pedestrians and vehicles accurately while the radar subsystem determines object positions and velocities accurately. By unifying these subsystems through sensor fusion, we improved object detection accuracy by over 10% (compared to vision alone). Additionally, we found that the radar and vision subsystems have different failure cases, which makes our unified system more robust (sunlight does not affect the radar, for example).

Based on our results, we **successfully developed a solution** to meet the aforementioned user needs by creating a custom standalone perception system that can function independently or in tandem with an existing system. Through the sensor fusion of stereo vision and radar technologies, our system identifies objects in most driving conditions, while still being inexpensive, compact, and efficient relative to current solutions.

Clearly, the **motivation for this project** stemmed from the desire to improve automotive perception for autonomous driving. To do so, we explored the use of complementary stereo-vision and radar sensor technologies to increase the accuracy and reliability of identifying objects in the driving environment.

2. Use case

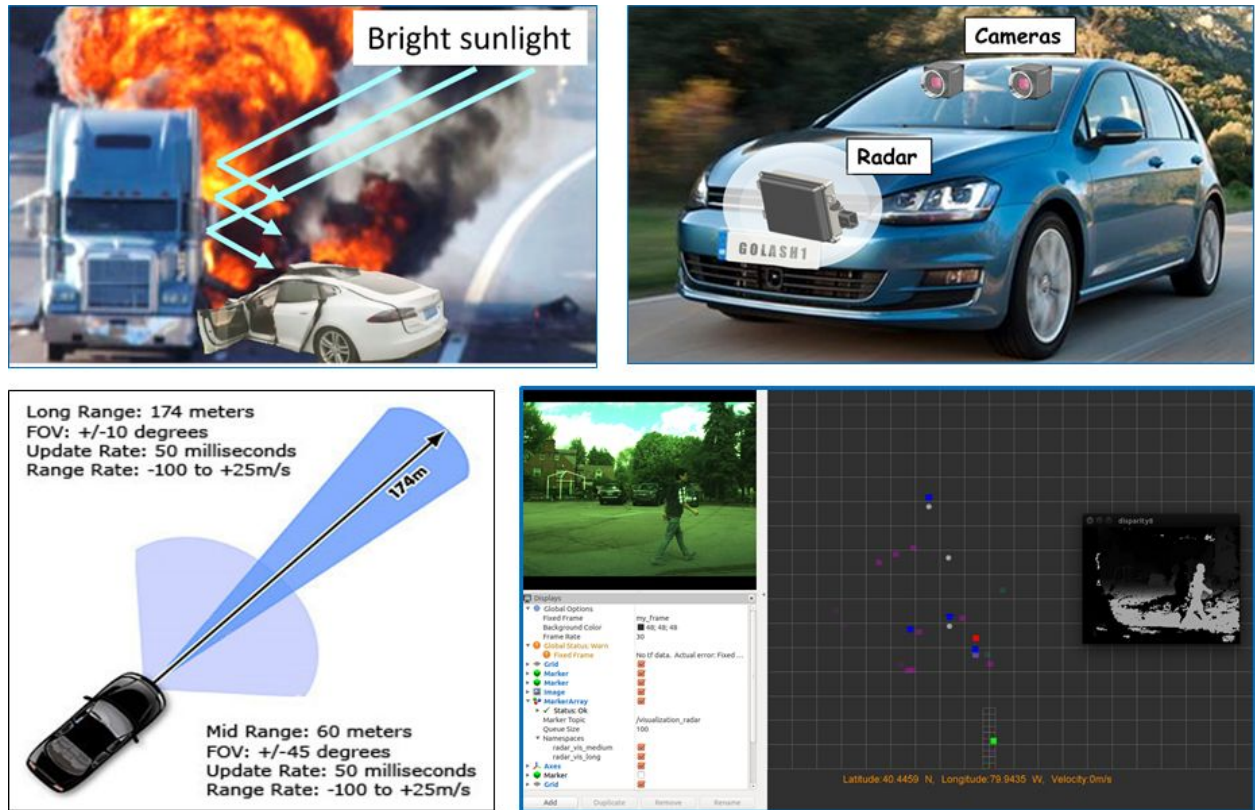


Fig. 3. Use case scenario images

Messla Motors Inc. recently lost one of their autonomous cars even though it was equipped with multiple redundant cameras and LIDAR sensors. This car crashed into a white truck when the sunlight reflected off the truck's side compromised the car's sensor system (top-left image in Fig. 3.). When Messla Motors' stock fell sharply, the CEO of the company, Mr. Dusk, was faced with scrapping their expensive sensor rack project and starting over. Moreover, recent consumer reports had shown that the bulkiness and the difficulty of maintenance were major downsides of the Messla Motors sensor array. The Messla Motors crash scared away many small entrepreneurs and aspiring researchers from venturing into the autonomous vehicle development field. They felt that if a big company like Messla Motors had failed to design a foolproof perception system, then there was no way for them to do so on their limited budget. Consequently, interest and progress in the autonomous vehicle research and development industry started to fall. What could be done about this?

Enter the SeeAll system from Team Aware! Our perception system uses a radar to simultaneously perceive long and short range objects (bottom-left image in Fig. 3). By combining radar and stereo vision technology, our unified system performs in cases where existing vision-based systems might fail, such as in bright sunlight. Our radar system detects all metal objects (vehicles, lamp posts, lane dividers etc.) up to 120 meters away and is unaffected by ambient lighting conditions. Additionally, our radar unit is about 8 times cheaper than a

LIDAR unit with similar range and rate capabilities. Under favorable conditions, the stereo vision system accurately identifies vehicles, pedestrians, and obstacles up to 30 meters away. The combined system can thus accurately detect and classify objects using vision and then track their positions and velocities using radar. Our system thus uses sensor fusion to increase our object detection accuracy and overall robustness. Since our perception system uses just three primary sensors, it is significantly cheaper than systems like Mr. Dusk's. Our system is designed to be standalone, and the form factor of our design makes for an adaptable perception system that could be used by any manufacturer of autonomous vehicles.

Mr. Dusk proceeded to scrap his old perception system and try out the SeeAll system from Team Aware instead. He easily installed the sensors into his test vehicle (as depicted in the top-right image in Fig. 3.) within minutes, and then connected the computer system to the car's display and control systems. He then eagerly entered the car to see our system perform! As the car drove around the city, the SeeAll system accurately identified and located objects of interest. A live video feed and with a grid showing object types, locations and velocities were displayed side-by-side on the infotainment screen (as shown in the bottom-right image in Fig. 3). The car's control system designed by Messla Motors easily read the relevant parameters from our perception system and maneuvered the car accordingly. During testing, it started to rain. This was not a problem for our system - the cameras were installed within the car's cabin and the radar is weatherproof. Later, the sun came out and shone brightly into the cameras. This too was not an issue for the SeeAll system, since the radar was unaffected. Just as the car turned back towards the testing facility, a crazy pedestrian dashed across the road in front of the car. Thanks to our real-time performance and our dual modes of perception, the pedestrian was detected, and the brakes were applied perfectly. Mr. Dusk was completely impressed.

Following Mr. Dusk's example, smaller entrepreneurs and researchers were emboldened to use the SeeAll sensor system for their projects and autonomous vehicles. Some chose to completely replace their existing system with the SeeAll, while others chose to augment their existing system by integrating our system into theirs. The stereo cameras and the radar work hand-in-hand to create a sensor system that is full-range and real-time. By using stereo vision for object detection and classification and then radar for velocity and position monitoring, we avoid using expensive components such as LIDARs. This means that our system is low-cost and can therefore be used with minimal risk by even small-scale research and development labs.

3. System requirements

Our system-level requirements were modified and updated during the progress of the project to be more realistic given our constraints and our initial results. A key change was the de-scoping of object tracking using stereo-vision. Our requirements are categorized into Performance (P) or Non-functional (N) requirements. Each requirement is further categorized as Mandatory (M) or Desirable (D) according to our project goals. The performance requirements and non-functional requirements can be found in the tables sections 3.3 and 3.4, respectively.

The four subsystems that make up our system can be listed as follows: the sensor mounting subsystem, the power supply subsystem, the sensor fusion subsystem, and the perception subsystem (as shown in Fig. 4). The target parameters and performance of each system and subsystem requirement are described below, followed by a summary of the changes in requirements we made over the course of the project.

3.1 Subsystem requirements

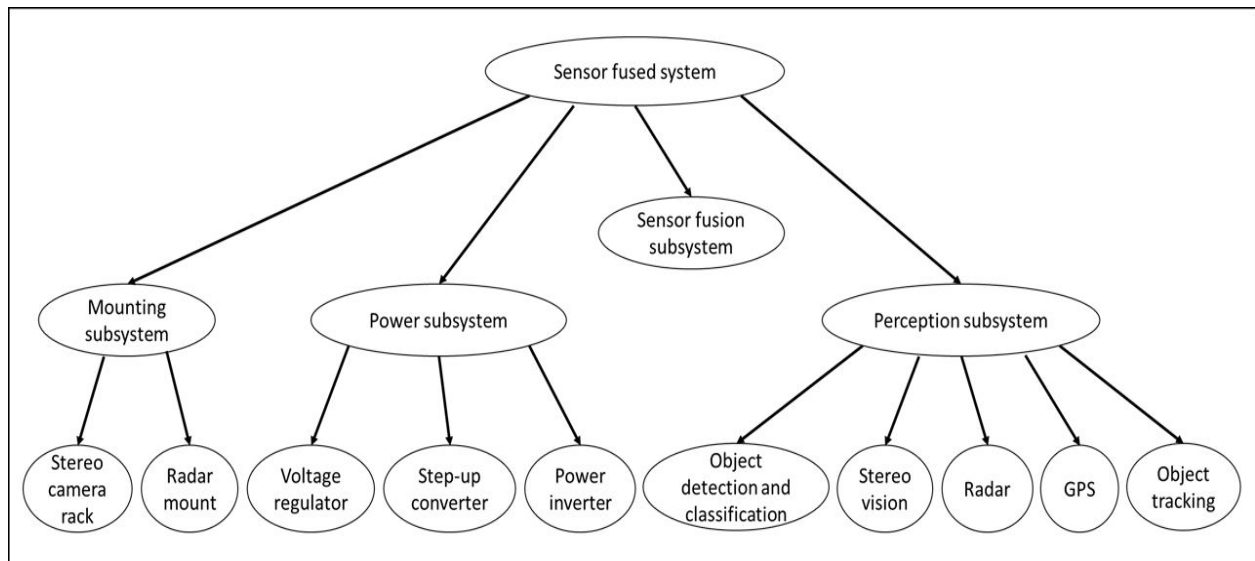


Fig. 4. The subsystems breakdown of our perception system

3.1.1 Sensor mounting subsystem

Radar mount: The Delphi electronically scanning radar (ESR) unit requires to be installed in the car such that the center of the radar sensor is between 30 cm and 86 cm above the road surface [3]. The mounting structure should be firmly attached to the test vehicle and should be robust in resisting the vibrations of the car. Additionally, the radar mounting solution must not obscure the view in front of the radar with any metal or non-uniform material.

Stereo camera rack: The two cameras need to be fixed at a constant known distance for stereo vision. The stereo pair should be firmly fixed inside the car using a rigid, lightweight mounting bar, at a baseline of 50~60 cm. This baseline for the stereo camera pair (including the relative pitch, yaw, and translation) should not change after driving for an extended period over varying road conditions. Additionally, the position and orientation of the cameras should not change relative to the radar sensor unit or the car's chassis. Obviously, this custom stereo setup will require that the entire field of view in front of the cameras is unobscured and clear once it is installed in the car.

3.1.2 Power Subsystem

Voltage regulator: A car battery's voltage output can vary from 12 to 15 VDC during driving. In order to prevent any sudden voltage changes from affecting our expensive sensors, it is required that our power subsystem use voltage regulators. To power both cameras, a 12 VDC regulator capable of 2 A (24 W) continuous is required. This voltage regulator should be efficient and compact so that it can be easily installed into the car.

DC-DC step-up converter: A DC-DC step-up converter module is required to boost the input from a car battery to a stable 24 VDC at at least 10 W to power the radar sensor unit. The power booster should take the 12 to 15 VDC car battery input and then convert and regulate it at 24 VDC for the radar to perform optimally.

Power inverter: The power inverter subsystem is required to power the computer. The output power must be at least 180 W to power our computer, since it has a high performance core i7 CPU and a Titan X GPU. The power inverter thus is required to connect to the car battery and convert it to 110 VAC at 60 Hz with at least 180 W continuous operation capability.

3.1.3 Sensor Fusion Subsystem

The sensor fusion subsystem is the software setup that must meaningfully combine the data from the radar and the stereo vision inputs. This subsystem must acquire object information data such as category, velocity, and position from the radar and stereo vision sensors. It must then synchronize, filter, and compile the data in order to create a global depth map that shows the positions and velocities of the surrounding objects relative to the test vehicle. It is required that the system synchronize the sensors, update calculations, and display the fused output to the user at a rate of at least 5 Hz for real-time performance. The sensor fusion subsystem should take its input from the perception subsystem and output the final processed result to a visualization that can be understood by the user (as shown in Fig. 5). The sensor-fused processed data should be in a format that can be read and integrated easily into an existing autonomous vehicle's control or perception system.

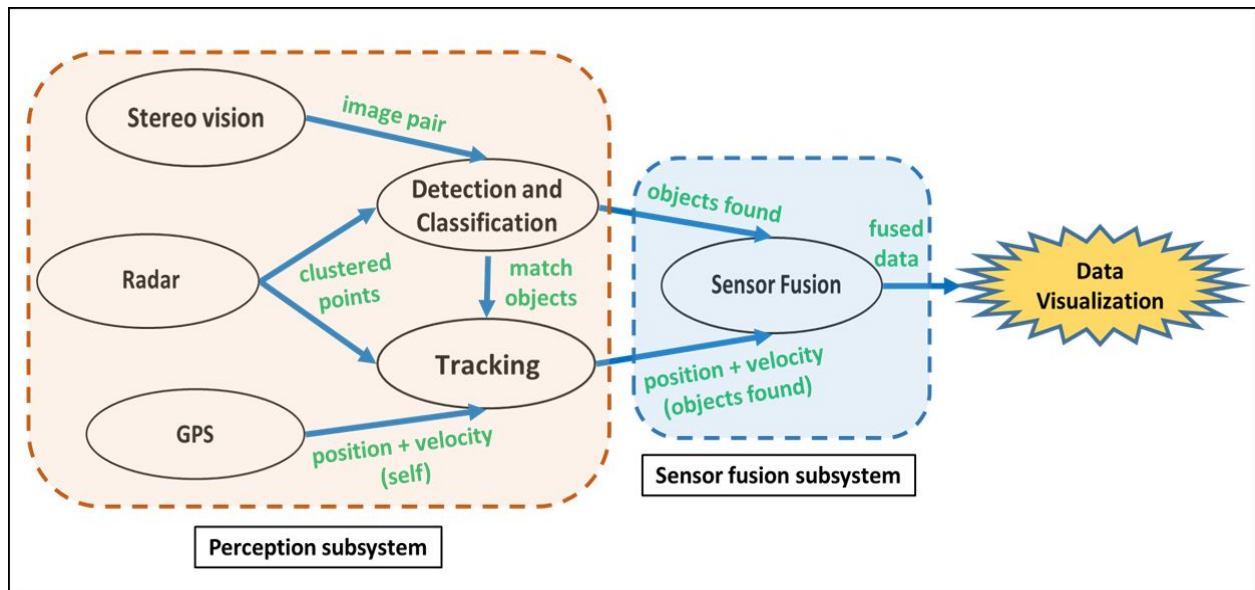


Fig. 5. The Perception and Sensor Fusion subsystems working together

3.1.4 Perception Subsystem

Object detection & classification: The object detection and classification subsystem must categorize the detected objects of interest into vehicles or pedestrians. This subsystem must take as input the images captured from the stereo vision system and then use efficient object detection and a depth-map-calculating algorithms to perceive the environment. It must also acquire the filtered detection-level data from the radar subsystem. The outputs of this subsystem to the sensor fusion subsystem should be the positions, classifications, and confidence scores of the detected objects. This subsystem should be robust in a variety of weather conditions (rain, fog, snow etc.) and different lighting environments (direct sunlight, headlights, night traffic etc.). Additionally, this subsystem must achieve at least 60% object detection accuracy, at least 80% object classification accuracy, and at least 80% object position estimation accuracy for objects up to 30 meters away.

Stereo vision: The stereo vision subsystem should take images captured from both cameras simultaneously as input. The cameras must be triggered at the same time using either a software or hardware trigger method. This subsystem should capture images from the camera at a frequency that matches our throughput rate / update rate of 5 Hz (so as to not waste memory), and then output those images to the object detection and classification subsystem. The cameras should both be configured to have the same settings (exposure, shutter speed, resolution etc.) so that the pictures taken have similar photographic properties. This will allow for optimum accuracy and speed in calculating the depth math and classifying the detected objects.

Radar: The radar subsystem must detect and provide full-range, accurate position information for all vehicles up to 120 meters away [1][2]. The radar subsystem should output the filtered and clustered detection-level data of the objects to the object detection and classification subsystem. The radar subsystem should acquire the data at 20 Hz or greater and should complete filtering and clustering at ~20 Hz. This subsystem should use a Kalman filter for noise reduction and clustering of data-points. This will provide accurate position (depth) and velocity information for the objects found by the radar.

Object tracking: The tracking subsystem must determine the velocities and positions of the detected objects relative to the test vehicle. It should acquire the vehicle's velocity and position from the GPS subsystem, and then use that information to track the vehicles and pedestrians by using the radar subsystem data as well as the sensor-fused data. This subsystem takes the output from the object detection and radar subsystems as its input. The system should be robust to object occlusion, and the accuracy of the calculated object positions and velocities should be over 80%.

3.2 Updates on system requirements

Reduced object detection and classification accuracy: Through experimentation, we determined that the state-of-the-art Single Shot MultiBox Detector (SSD) algorithm was our best choice overall for detecting and classifying vehicles and pedestrians using our stereo vision subsystem. However, even with this algorithm, the final object detection accuracy was only around 70% for our setup. Additionally, the SSD algorithm is known to perform poorly on small objects. According its author, this is an unsolved issue and so it could lead to decreased performance in our final setup. We therefore realized that our previously desired accuracy of 80% was unrealistic. Instead, we sought to consistently achieve an object detection and classification accuracy above 60% in our final system (>60% using only vision, >70% using sensor fusion).

Descoped determining object sizes: We found that it was hard to reliably differentiate between the sizes of objects based on the size of their bounding boxes provided by the stereo vision subsystem. Technically, the radar would have the ability to determine relative size of objects based on the amplitude of detection. However, it could not be well verified through the limited testing that the team had conducted towards the end of the project. We thus decided that accurately finding the sizes of detected objects was not within the scope of this project. Our perception system instead focuses on accurately determining object types, positions, and velocities.

Descoped tracking capabilities: Even without the implementation of a vision-based tracking method, the stereo vision update rate hovers just above 5 Hz. We tested some fast multi-object tracking algorithms for stereo vision, but our system slowed down too much to be considered real-time. Since it is crucial that our system achieves real-time perception, we decided to descope vision-based tracking. Instead, we decided to use the radar subsystem for tracking object positions and velocities, since we found that the radar subsystem outperformed the stereo vision subsystem both in terms of accuracy and speed. Additionally, we descoped the implementation

of predictive tracking capabilities (interpolation of the car's path in the immediate future) in our perception system due to time constraints.

3.3 Performance requirements

Our system's performance requirements are categorized as Mandatory (M) or Desirable (D) according to our project definition and goals (Table 1). These requirements have been updated since the start of the project. As mentioned in the earlier sections, we had to slightly reduce the scope of the project and therefore realistically reassess our performance requirements. Fusion of stereo vision and radar data should work in the short range, up to ~30 meters away. Objects in this proximity should be detected by both sensor subsystems, while full-range detection and tracking of vehicles should be done by the radar subsystem. To allow for real-time performance, the data from all the sensors must be synchronized with each other.

Table 1. Performance requirement details

ID	Description
M.P1	The system shall detect and classify pedestrians within at least 20 meters
M.P2	The system shall detect and classify vehicles within at least 50 meters
M.P3	The system shall acquire raw data from all sensors at 5 Hz or higher
M.P4	The system shall detect objects with an accuracy of at least 50%
M.P5	The system shall classify objects with an accuracy of at least 60%
M.P6	The system shall synchronize the sensors to acquire inputs within 20 ms of each other.
D.P6	The system shall find the position and velocity of the test car with an accuracy of at least 80%
D.P7	The system shall find positions and velocities of objects with an accuracy of at least 80%

3.4 Non-functional requirements

The non-functional requirements of our perception system are also categorized as Mandatory (M) or Desirable (D), as shown below in Table 2. These requirements have been updated since the start of the project, although there have not been significant changes to them.

Table 2. Non-functional requirements details

ID	Description
M.N1	The system will work in real-time
D.N1	The system will not be destroyed by adverse weather conditions (rain, fog, snow etc.)
D.N2	The system will perform in all human-drivable lighting conditions
D.N3	The system will be concealable within the vehicle body
D.N4	The system can be integrated into existing vehicle display and infotainment systems

4. Functional Architecture

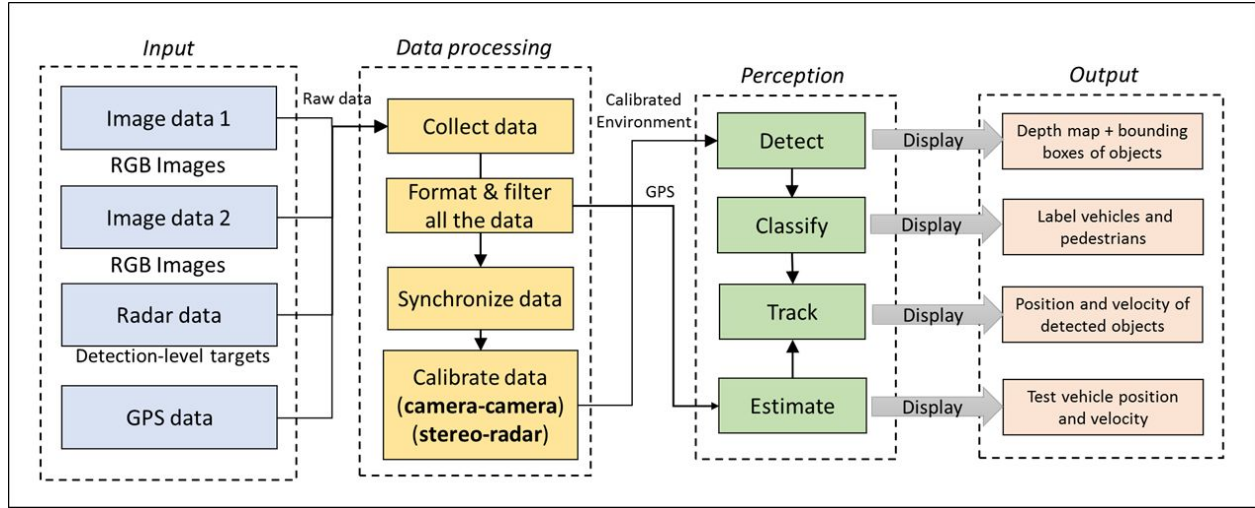


Fig. 6. Functional Architecture

The basic structure of our functional architecture has not changed significantly over the course of this project. A slight change was made to the format of the input data from the radar. The advanced built-in DSP firmware in the Delphi ESR 2.5 radar unit pre-clusters detected points into tracking targets. However, we found that the in-built clustering method was unreliable and not useful for real-world object tracking [3]. Instead, we choose to obtain detection-level data points from the radar directly and then process them using a Kalman-based filter for object tracking. We found that this method of determining object positions and velocities is more reliable and accurate.

A diagram illustrating the updated functional architecture of our system is shown in Fig. 6 above. Raw data from all our sensors are processed and then used to perceive the vehicle's surroundings. This process is mainly composed of two steps: data processing, and perception. These steps are explained below.

4.1 Data processing

Our custom stereo vision system uses two identical cameras which are mounted on an aluminum beam at a known baseline distance. We decided to build our own stereo vision setup since we felt that the existing solutions that we could afford did not have the required range or resolution. Our two single-lens, global-shutter CCD cameras are synchronized through the use of either a software or hardware trigger method.

We found that both hardware and software triggering mechanisms work equally well as long as the code runs consistently and without hiccups. When the cameras are triggered, they capture a pair of images taken simultaneously. The streams of data from the radar and the GPS sensors are first parsed to extract the relevant variables and then formatted into a readable,

understandable format. The different sets of data need to be synchronized with each other with respect to time. This is crucial, since specific applications of our system have shown us the importance of using real-time data for reliability. Processed data from the stereo camera and the radar are fused to improve the performance of object detection within 30 meters.

4.2 Perception

In the perception module, tasks can be roughly divided into four categories: detection, classification, tracking, and motion estimation of the test vehicle. Using the processed data as input, these tasks will be executed in the perception module as demonstrated in Fig. 6. The order of the tasks is determined based on their dependencies and prerequisites.

The first task is to detect objects of interest in the scene using the stereo vision data. Second, the detected objects are classified as either vehicles or pedestrians. The system distinguishes between vehicles and pedestrians by extracting features from images. Vehicles are further classified by type - car, bus, truck etc. Labels of these classified objects are displayed in the bounding box video output of our perception system.

Third, objects that are detected by the stereo vision and radar subsystems are correlated. We use a GPS module to determine our test vehicle's position and velocity. Matching objects that are found in both subsystems can then have their positions and velocities calculated accurately from the radar and GPS data. By using Multiple Hypothesis Tracking for sensor fusion of the objects detected by both radar and vision, we improve the overall object detection accuracy. We had initially planned to also include object size calculation and predictive tracking of objects in this project; however, we later descoped these features due to time constraints.

The final task involves visualizing the processed and sensor-fused data from our perception system. We display the results from the previous tasks in a meaningful way to the user by showing the detected objects in a 2-D grid that represents the driving plane (see bottom-right image of Fig. 3). In addition to displaying the detected objects and their properties, this task involves calculating and presenting the position and velocity of our test car as well (using the processed GPS data).

In summary, raw input data from all the sensors are processed first and synchronized, then a 2-D virtual environment (grid) is recreated based on the combined real-world data. In this environment, objects are detected and classified with labels. Their kinematic and spatial parameters of the objects and the test vehicle are also included. The representation of the driving environment created using the processed and unified data is the final output of our perception system. All the variables can also be used as input by an onboard computer to control the vehicle or to display custom information through the vehicle's user interface.

5. System-level trade studies

Over the course of this project, our team had to make a number of informed decisions in order to select optimal sensors, hardware components, and algorithms for our perception system. Often, a clear “best solution” was not readily apparent. In each case, we found that conducting a trade study about our options can be a highly effective and logical method of quantitatively assessing the pros and cons of each available option. We find that our trade studies fall into three categories based on the subsystems they affect. The trade studies in the following subsections of this report concern the vision subsystem, the radar subsystem, and the computing subsystem. These three subsystems make up the foundation of our perception system, and so decisions made regarding these subsystems directly affected the final outcome of the project. This further emphasizes the importance of conducting trade studies during the decision-making process.

5.1 Vision subsystem

We chose between cameras that use global shutters, cameras that use rolling shutters, and RGB-D cameras. RGB-D cameras typically have rolling shutters, but we considered them separately due to their valuable ability to sense depth in addition to picture. The trade study in Table 3 considers six attributes for these types of cameras that vary in importance on a scale of 1 to 5 based on our desired system attributes.

Table 3. Camera trade study details

Attribute	Importance (1-5)	Global shutter	Rolling shutter	RGB-D
Cost	4	2	5	3
Performance at high speed	5	5	2	3
Ease of use	3	4	3	2
Stereo performance	5	5	3	1
Field of view	4	3	2	5
Noise rejection	5	4	2	1
Total		102	72	63

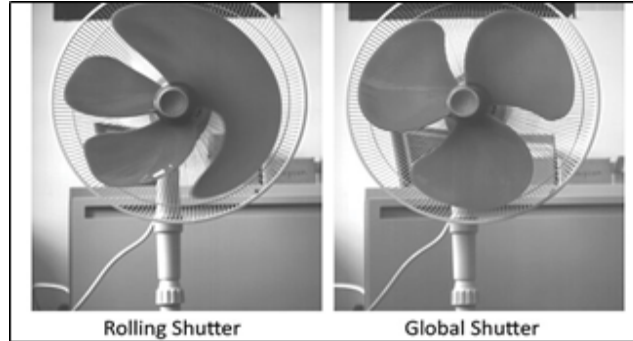


Fig. 7. Rolling shutter vs. global shutter

Although global shutter cameras are expensive, they capture the scene instantly. This prevents warping and edge distortions like the ones shown in Fig. 7. Such artifacts occur in rolling shutter cameras at high speeds. RGB-D cameras may allow for a wider field of view, but are limited by their range of depth sensing. Additionally, stereo RGB-D is challenging to implement due to IR interference between the two camera systems. Based on the trade study in Table 3, we chose to build our own stereo-vision system using two identical global shutter cameras. Our final system uses two 3.2 MP Color Grasshopper 3 cameras from PointGrey. This camera uses a global shutter and has an appealing performance-to-price ratio.

5.2 Radar subsystem

This trade study compares the pros and cons of various perception technologies. To sense depth independently of stereo vision, we chose between radar, lidar, and ultrasonic (sonar) sensor systems as shown in the trade study in Table 4 on the next page.

Table 4. Depth sensor trade study details

Attribute	Importance (1-5)	Radar	Sonar	Lidar
Cost	5	3	5	2
Light sensitivity	5	5	5	3
Dust/fog sensitivity	5	4	3	2
Ease of maintenance	4	5	4	3
Rate of sensing	4	3	1	5
Field of view	4	2	3	5
Size of sensor	3	4	4	3
Ease of use	3	4	2	3
Range	5	4	1	5
Total		144	120	130

For the same levels of performance, lidars are more expensive than radars (by almost a factor of 8) and large lidar units can be hard to maintain due to moving parts. Additionally, they are harder to conceal within the car's body due to their method of operation and their typically larger form factor. On the other hand, lidar sensors can detect shape information while radar can only detect position. Sonar is a lot slower than both radar and lidar, and hence would not be a wise choice for an autonomous vehicle. Our radar system can function effectively in various lighting conditions, it has no moving parts, and it has a small form factor. However, it has a narrower field of view and a slightly shorter range compared to some lidar sensors. In the end, we chose to use a radar system in conjunction with a custom-built stereo vision system. We determined that this was the optimal solution given our needs and requirements.

5.3 Computing subsystem

The computing subsystem consists of the processing hardware and software. We used our budget to buy a powerful desktop computer so that our system's computational speed would not be an issue down the line. We did not want slow processing power to prevent our system from perceiving the environment in real-time. We found that the algorithm that takes the longest time to compute for each new set of data was the depth-map calculation algorithm in the stereo-vision subsystem. It was thus imperative for us to carefully choose an algorithm that could perform in real-time without sacrificing on the quality of the stereo-vision depth-map. Table 5 covers the trade study we conducted between the SGBM (Semi-Global Block Matching) and ELAS (Efficient Large-Scale Stereo Matching) algorithms for stereo matching.

Table 5. Stereo-vision algorithm trade study details

Attribute	Importance (1-5)	SGBM	ELAS
Speed (fps)	5	5	1
Ease of integration	3	3	3
Quality of map (density)	2	2	4
Complexity	2	2	2
Available support	5	5	2
Total		67	36

In general, the SGBM algorithm performs much faster than ELAS. During our tests on a conventional laptop, ELAS ran about 1 fps while SGBM could run up to 5 fps (both on ROS). This is mainly because that ELAS is implemented on a single thread. However, according to our experiments, ELAS could provide a denser disparity map. With all things considered, we decided to use SGBM in our stereo vision subsystem, aiming to achieve the higher processing speed by paying the cost of lower quality of the disparity map. The specific attributes that supported our reasoning and helped us make the final decision is listed in the Table 5 above.

6. Cyberphysical architecture

There are two major parts to the cyber-physical architecture: the hardware part and the software part. The cyberphysical architecture of our system is outlined in Fig. 8 below. Due to our project's focus on building a perception system, we gave most of our attention to the software aspect. All the black arrows in the figure represent information flow between sensors and software modules. The arrow from power subsystem represents the flow of energy. In our project, the input is the raw data from our sensors, and the output is a virtual reconstruction of the driving environment around our test vehicle based on our processed data.

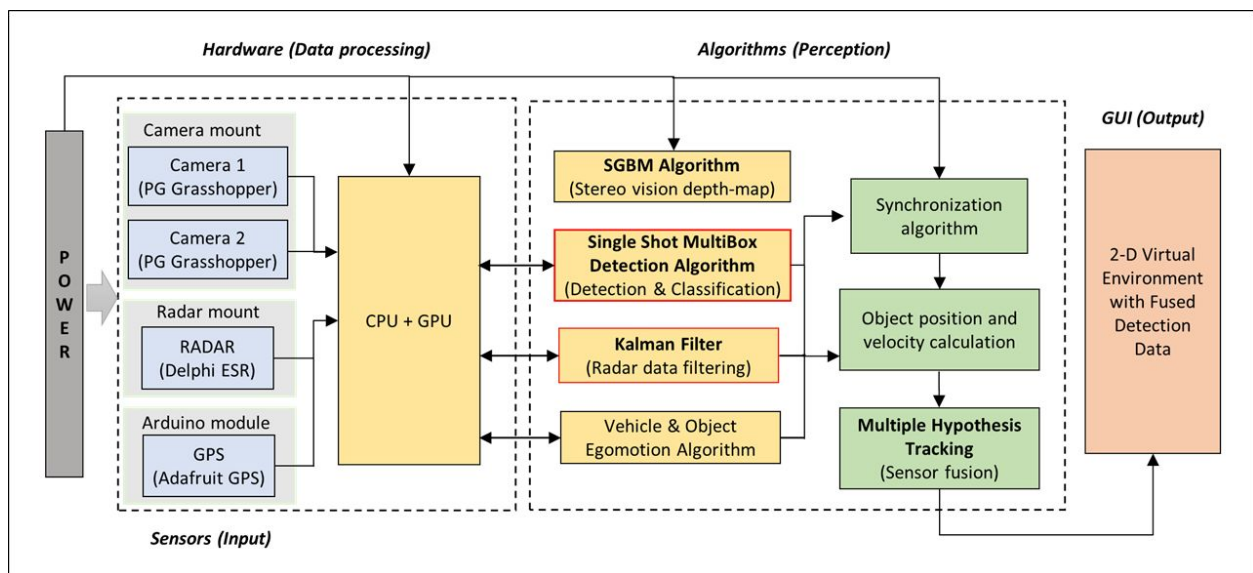


Fig. 8. Cyber-physical Architecture

6.1 Hardware

The hardware section, represented on the left in Fig. 8, contains the power source, the sensors and the computer. The power source supplies DC power to the sensors and AC power to the computer (using a DC-AC power inverter). Different power sources were used during the course of the project. For indoor testing, we relied on a 30 V, 5 A DC power generator connected to the AC mains power supply. For testing the sensors on-board our car, we initially connected a small, 150-watt DC-AC inverter to a cigarette lighter socket of the car. Our power requirements grew when we decided to use a power computer for this project, and so the cigarette lighter socket was not a viable option (it could supply at most 120 W of power) later on in the project. We eventually decided to use a 1100 W DC-AC inverter connected to the car's battery to power the computer. A 12 VDC voltage regulator and a 24 VDC step-up converter were also connected to the battery to power the cameras and the radar, respectively.

There are three sensor subsystems that make up our final system. We mounted a Delphi ESR 2.5 radar unit to the front of our car. We built a custom stereo vision subsystem composed of two identical cameras (PointGrey Grasshopper 3), which are mounted in the testing vehicle at a fixed baseline distance. Lastly, we use an Adafruit Ultimate GPS sensor connected to the computer via an Arduino Micro. In addition to providing our system with GPS data, the Arduino module also enables hardware triggering of the cameras via their GPIO pins. These three sensor subsystems together collect information about detected objects in the vehicle's environment in real-time. All the raw data acquired is passed to the software perception module, where filtering, synchronization, and interpretation tasks take place. The processing unit essentially serves as the center of all communication for our standalone perception system.

6.2 Software

The software section, represented on the right in Fig. 8, contains the algorithms and methods used to collect, store, and process the data flowing in and out of our system. The CPU of our computer is programmed to run all our calculations except for the stereo-vision object detection and classification. The SGBM algorithm runs on the CPU and is used to calculate the stereo-vision depth-map. Stereo-vision-based object detection and classification is done by running the SSD algorithm (Single Shot MultiBox Detector) on our Titan X GPU using CUDA. Our various algorithms yield processed data that together contain all the information needed to create the virtual environment. Object positions and velocities are computed independently by the vision subsystem and the radar subsystem. These values are correlated between subsystems (and sensor fusion is thus achieved) by using Multiple Hypothesis Tracking (MHT) on the detected points, based on position.

For object detection and classification, we use the state-of-the-art Single Shot MultiBox Detector (SSD) algorithm on the left camera image. The SSD algorithm involves the use of a deep neural network, which is why it achieves both object detection as well as object classification at high speeds with high accuracy. Based on experimentation, we found this algorithm to have a relatively good balance between performance (detection rate and accuracy) and speed (number of frames processed per second).

Aspects of the cyber-physical architecture have been updated many times since the start of the project to reflect new system developments and key decisions. For example, we decided to use a GPS sensor unit for our position and velocity ground truth. Initially we had considered reading that data from the car's OBD port or from an IMU sensor instead. As mentioned earlier, we found that the pre-clustered targeting data that obtained from the radar directly was too noisy to be useable. Instead, we chose to use our own Kalman-based filter on the detection-level object positions obtained from the radar, because we found that this was a much more reliable method for detecting and tracking objects using the radar. Lastly, we found that although our stereo-vision subsystem could reliably identify objects within ~30 meters, it could not accurately determine their positions or their velocities. We thus included a position-based Multiple Hypothesis Tracking (MHT) algorithm for sensor fusion. This algorithm uses the filtered radar data to improve overall object detection accuracy by over 10% (compared to solely vision-based object detection).

7. System description and evaluation

In this section of the report, we describe and discuss the details of our system and its subsystems. We look at how the system evolved during the course of the project, and we reflect on the final features and test results of the perception system that we developed.

7.1 Subsystem descriptions

Our system can be broken down into three essential subsystems: the vision subsystem, the radar subsystem, and the computing subsystem. Each of these subsystems can further be divided into their hardware and software components. Through the linking and collaboration of these subsystems, data from all the sensors are processed and combined.

7.1.1 Vision subsystem

This subsystem includes the two 3.2 MP Color CCD PointGrey Grasshopper 3 cameras used for our stereo-vision setup. It also includes the mounting solution for the cameras. This consists of the 8020 aluminum beam onto which the cameras are mounted, the 3D-printed camera housings, and the modified sun-visor mounts using which the rack is installed in the car (all this can be seen in the images in Fig. 9 below).



Fig. 9. The vision subsystem consists of the cameras and their mounting solution

We can consider the Arduino module to be part of this subsystem as well. This is because the Arduino module consists of an in-line 12 VDC voltage regulator that also regulates the power to the cameras. Moreover, in addition to the voltage regulation for the cameras, this module can do two things: 1. hardware trigger the cameras, and 2. communicate GPS data to the computer. The improved camera wiring can be seen in Fig. 10 below, which includes the addition of a voltage regulator and an Arduino Micro onto the same power circuit as the cameras. The Arduino Micro reads GPS data from the Adafruit Ultimate GPS sensor module at a rate of 5 Hz and sends it via serial port to the computer for processing. The Adafruit GPS sensor has a high

sensitivity of -165 dBm and it supports 66 channels. It updates at a rate of up to 10 Hz while consuming very low power, making it apt for use in our project.



Fig. 10. (Left) The Arduino Module box includes a 12 VDC voltage regulator and an Arduino Micro. The plug on the left connects to the camera GPIO pins; the one on the right is an HXT connector for the battery. (Right) The Adafruit Ultimate GPS sensor and its antenna are designed to work with Arduino boards.

Results: The camera mounting solution proved to be robust and reliable, since the cameras were found to not move any measurable distance relative to each other and relative to the car, even after testing on uneven roads. The hardware trigger method using the Arduino module works well to synchronize the cameras, and this was tested by taking pictures of a stopwatch. The voltage regulator performed well, and the cameras never lost power or overheated during testing. The quality of the GPS data acquired was significantly improved by connecting the matching external antenna to the Adafruit GPS sensor. Despite this, we found that the GPS-based values obtained for the vehicle angle and velocity were unreliable and sometimes noisy. Another downside of this system is the ineffective shielding of the cameras from reflections off the dashboard. This could potentially result in reduced performance.

7.1.2 Radar subsystem

This subsystem includes the Delphi ESR 2.5 radar unit and our mounting solution for installing it onto the test car (left image in Fig. 10). It also includes the DC-to-DC step-up converter that provides a stable 24 VDC to power the radar unit, using the car battery as a source (right image in Fig. 10). The ESR 2.5 radar unit has been used successfully by Delphi for automotive perception and autonomous driving applications. This radar sensor simultaneously detects close range and long range objects in the driving environment at a rate of 20 Hz, as shown in Fig. 11. We can connect to the radar sensor using either CAN bus or ethernet. As mentioned earlier, the radar sends tracking-level data via the CAN bus and it sends detection-level (raw) data via ethernet.

Results: The final radar mounting solution that we developed is shown in Fig. 12 below. The radar unit is designed by Delphi to be weatherproof, so we directly mounted it onto the grille of the car using 3D-printed mounts. We were careful not to obstruct the sensor with any metal or non-uniform surface. After failing to power our sensors in the car using our power distribution PCB, we switched to an off-the-shelf converter. This converter proved to be robust and it was

able to consistently power the radar unit without overheating or voltage jumps. Unfortunately, we found that the data from the CAN bus (the tracking data computed by the radar's built-in clustering method) proved to be futile for tracking purposes due to high noise and error. We later found a Kalman-based filtering method for the detection-level data points acquired via ethernet. This allowed us to accurately determine the positions and velocities of detected targets.

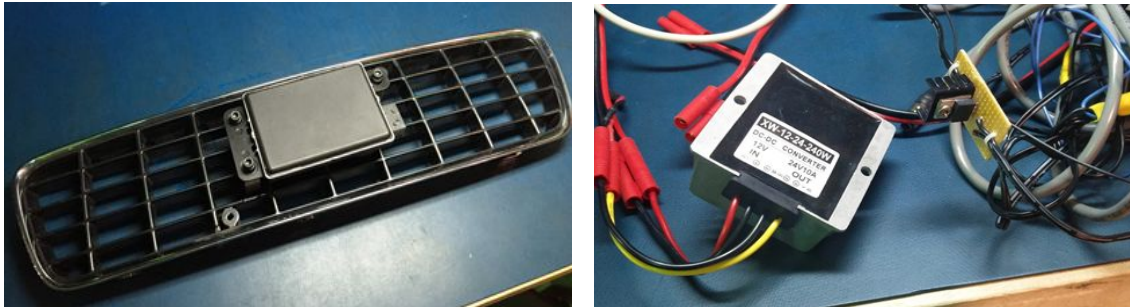


Fig. 10. (Left) The radar unit attaches to the plastic grille of the car using 3D-printed mounts we made. (Right) The step-up converter on the left and the voltage regulator IC on the right.

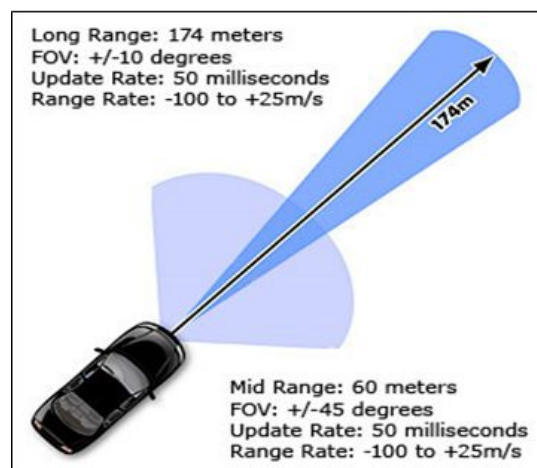


Fig. 11. Performance specifications of Delphi's ESR 2.5 radar sensor unit



Fig. 12. The radar is mounted on the grille, while the stereo-vision setup is installed inside the car cabin.

7.1.3 Computing subsystem

This is the most important subsystem, as it is the one that is responsible for determining the properties of the driving environment by processing data from all the sensors. This system consists of the computer and its hardware components, but more importantly it consists of all the algorithms and software methods used to make our system perform.

Hardware: We used a large part of our budget to invest in a desktop computer for this project. Despite using advanced optimized algorithms to increase processing speed, we still needed a computer with a powerful CPU and GPU for our system to update the reconstructed virtual driving environment in real-time. Hence, we ordered a computer equipped with a powerful Intel i7 CPU and the Nvidia Titan X GPU.

Software: Even though we had acquired a powerful computer, we needed to be thoughtful when choosing algorithms for our system. Without selecting fast algorithms, there was no way we could achieve real-time performance. The software aspect of our computing subsystem therefore depends on these key modules and algorithms:

1. The SGBM algorithm: This algorithm runs on the CPU and is used to calculate the disparity map using the stereo-vision image pairs. This algorithm runs the slowest relative to the others, and it therefore limits the overall system speed. An example disparity map calculated by this algorithm is shown in Fig. 13 below, with a reference image taken from one of the cameras.

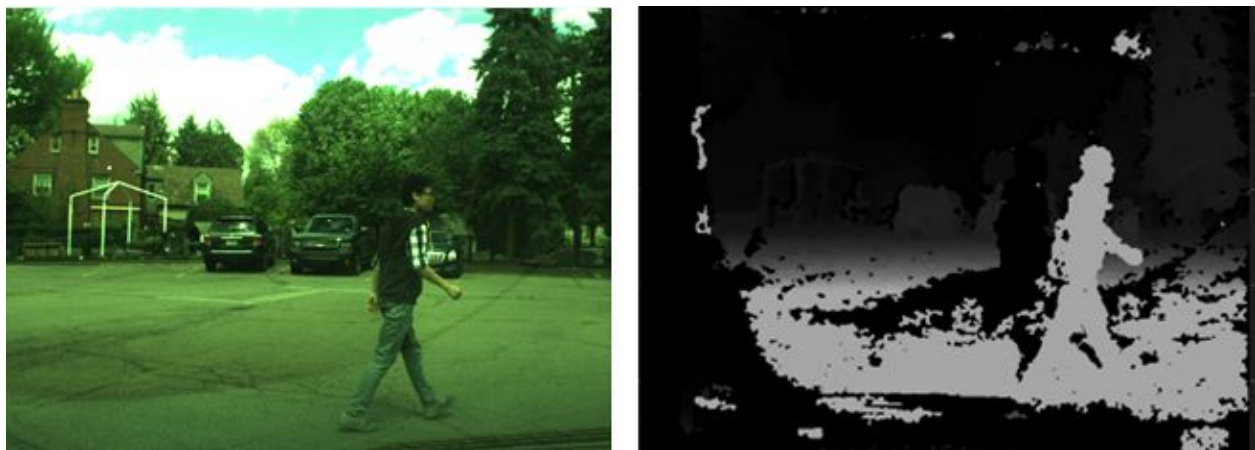


Fig. 13. (Left) Reference image from one of the cameras. (Right) The disparity map computed by SGBM.

2. The SSD algorithm: This algorithm runs on the GPU using CUDA. It is used for object detection and classification. Through the use of a deep neural network, detection as well as classification of objects are computed efficiently and quickly. The output of this algorithm is the original image plus bounding boxes and labels for the objects identified. By correlating the output of this algorithm with the disparity map obtained by the SGBM algorithm, object positions can be determined using stereo-vision. We can see an example output of this algorithm in Fig. 14 below.

4. Multiple Hypothesis Tracking (MHT): In order to correlate the parameters of objects detected by both stereo vision and radar, we implement position-based MHT. Thanks to the radar's ability to accurately detect and locate vehicles (and other metal objects), we can use sensor fusion done using MHT to boost the system's overall object detection accuracy. We found that, on average, sensor fusion using position-based MHT resulted in at least a 10% increase in object detection accuracy relative to solely vision-based object detection. This figure is higher when only vehicles are considered.
5. Synchronization and display: Our system uses ROS as the framework for synchronizing sensor data and processing and displaying it. Since our sensors provide data at varying frequencies with varying time lags, we match timestamps in order to synchronize data with respect to time. Our final output of a virtual environment is displayed using rviz in ROS, as shown in Fig. 15 above. The architecture of our ROS setup is explained by the depiction in Fig. 16 below.

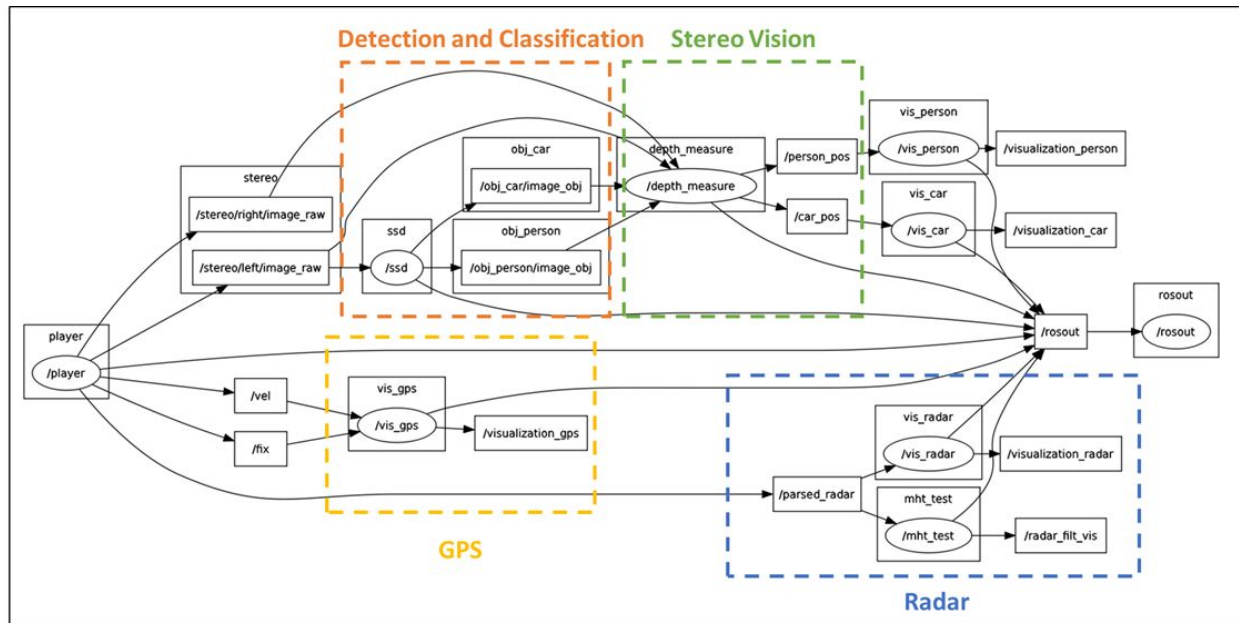


Fig. 16. The ROS setup that our software platform is based on.

7.2 Modeling, Analysis, and Testing

This section summarizes some of the modeling, analysis, and testing of system components and software modules over the course of the project. The following tasks are of importance:

1. *All the components of our sensor mounting solutions for our perception system have been modeled using the Computer Aided Design software (SolidWorks).* The camera housings and the radar mounts were modeled by carefully examining the dimensions of the sensors and the space available within the testing vehicle. The CAD models were then fabricated by 3D-printing and tested for form and function in the vehicle. After much experimentation, the final versions were 3D-printed and used to mount the two cameras and the radar onto the testing vehicle. The final CAD models are shown in Fig. 17 below.

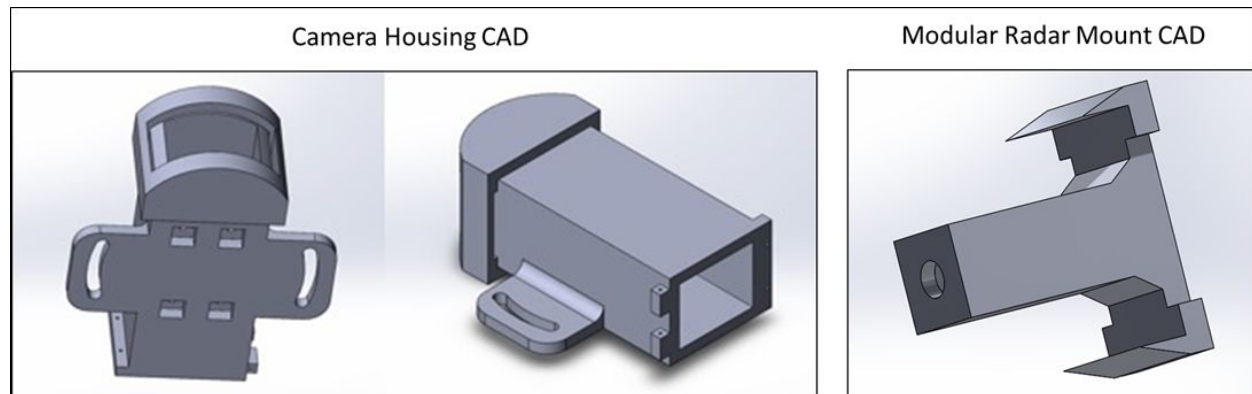


Fig. 17. The finalized CAD models for the camera housings and the radar mounting brackets.

2. *The baseline for our stereo vision system was determined through calculation and experiments.* These experiments were conducted in the corridor outside the lab. The two cameras were mounted at an appropriate, fixed height and orientation, similar to how they are mounted in the testing vehicle. According to our requirements, our stereo vision system should be able to detect objects of interest up to about 30 meters in front of the vehicle. Therefore, a checkerboard of appropriate size was placed at 30 meters in front of the vehicle as the target. In each experiment, the two cameras were fixed at a known baseline and the stereo vision system was calibrated using the Stereo Vision Toolbox in MATLAB. Then, the depth of the target (checkerboard) was calculated using the triangulation method, and the accuracy of the stereo depth calculation is determined. The data collected from all the experiments were then compared and analyzed. The final result suggested a baseline of 60 to 80 centimeters in order to achieve ~80% accuracy in the position estimation of objects within 30 meters.

3. *Several on-road tests were conducted to verify effectiveness of mounting design and basic sensor functionalities.* After installing and setting up all the sensors in the car, we measured their distances from each other and parts of the car's chassis. We then tested the basic functionality of the sensors by recording data as we drove our test vehicle around town. The testing weather conditions included light snow, moderate rain, sunset (when sunlight falls directly on the camera's lens), and nighttime. We found that the cameras were able to capture high-resolution

pictures that could be used to for object detection in most of the cases, despite the unideal weather conditions. The radar was unaffected by the different environment conditions and was able to collect data in real-time. After real-world testing was complete, the relative positions of the sensors was measured again. We found that the sensors had not moved by any measureable amount, thus verifying the quality of our mounting methods.

4. *Performance of the object detection and classification was analyzed using the testing data.* After testing basic sensor functionality and verifying that our mounting methods are robust, we collected multiple datasets to test our system's object detection and depth calculation capabilities. We found that we could achieve an object detection and classification accuracy of over 60% using the pictures captured by the cameras. These results satisfy the performance requirements of our stereo vision system. The radar was tested as well and it was found to determine object positions with an accuracy of over 80% (using a Kalman-based filter on the detection-level radar data).

5. *Performance of the power distribution PCB was analyzed and an off-the-shelf solution was chosen instead.* We tried to test our PDS PCB under small loads of up to 2 A at 24 W. However, after numerous failures, including the PCB getting burnt, we decided to use a proprietary solution instead. We arrived at this decision so as to prevent damage to our expensive sensors. The final power distribution circuit utilized a 12 VDC 5 A voltage regulator and a 24 VDC 10 A step-up converter connected to the car's battery to power all the sensors. The computer was powered using a high-power DC to AC inverter connected to the battery.

6. *Various solutions were tested to reduce the amount of light reflected into the cameras.* In some scenarios, the images captured by the cameras were found to have reflection artifacts in them. We found that bright overhead lighting, dust on the inside of the windshield, and a smooth or shiny car dashboard were leading causes of this. To reduce the risk of confusing our stereo-vision subsystem with reflections, we moved the cameras farther back into their housings, and we placed a fuzzy blanket on the dashboard to diffuse the incident light.

7. *Camera synchronization was tested using a stopwatch.* To test the reliability of our hardware and software trigger methods, we pointed our stereo-vision sensor rack towards a laptop screen that was running a stopwatch and then triggered the cameras simultaneously to see if the same times were captured in the corresponding image pairs taken by the cameras. In every pair of images, matching times were displayed on the screen. We thus determined that both hardware and software triggering of the cameras work well for the purposes of synchronization. However, we noted that software triggering is only as reliable as the execution of the code is. Given the complexity of our system's software, we feel it is safer to rely on a hardware trigger method that is independent of the processing setup.

7.3 SVE Performance evaluation

Although we were able to successfully complete most of the planned tasks we had set out to do by the SVE (see Table 6), we were unable to meet all the criteria. In the SVE, we did not include the velocities for the detected objects. This was due to simple oversight; we did have the system capabilities to calculate and display the velocities at the time. For the SVE Encore, we were more careful about presenting all the features of our perception system, and so we made sure to demonstrate them.

Even though we can claim that we have working implementations for a variety of advanced perception methods, many of our implementations need further work and tuning in order to perform consistently. We need to modify the algorithms and methods we currently use to be better suited to our system and applications.

Table 6. SVE performance checklist

Success Criteria	SVE	SVE Encore
Tracking ID of each object displayed in GUI	No	Yes
Relative position of each object displayed in GUI	Yes	Yes
Velocity of each object displayed in GUI	No	Yes
Classification of each object displayed in GUI	Yes	Yes
Absolute position (Lat. & Long.) of host vehicle in GUI	Yes	Yes
Absolute velocity of host vehicle in GUI	Yes	Yes
Accuracy of filtered depth value of objects > 70%	No	Yes
Accuracy of detection and classification by vision > 60%	Yes	Yes
Accuracy of detection by integrated system > 70%	No	Yes

7.4 Strong and weak points

The following strengths and weaknesses of our perception system were noted by our team over the course of this project. The strong points are what we depend on for our system's performance, whereas weak points are potential issues we might be able to fix in the future.

Strong points:

- Robustness of sensor mounts - Our sensor mounting system is robust. After repeated outdoor driving tests in various road and weather conditions, we noted that the positions of the sensors stayed the same. The effectiveness of our mounting solutions provide a strong foundation for the on-road performance of our perception functions.
- Object classification accuracy - The object classification accuracy is above 80% for detected objects, which exceeds our expectations.
- Radar position estimation accuracy - Our radar system provides depth information for objects of interest with an error rate of less than 5%. We use sensor fusion to thus bolster the performance of our stereo-vision subsystem.
- Powerful computer: We selected high-end components for our project computer. As a result, it can perform calculations very fast and allow us to perceive the environment in real time.

Weak points:

- Noisy tracking-level data from the radar - The radar gives us terribly noisy data when we try to acquire tracking data via the CAN bus. This is the case even if the testing environment is an empty garage. For now, we have our own tracking and filtering method that works well. However, in the future we would like to try and use these automatically calculated tracking points, provided we can extract useful information.
- Stereo vision disparity map - Calculating the stereo-vision disparity map takes longer than we would like. Right now, we run the SGBM algorithm on our computer's CPU for a refresh rate of ~5 Hz. In order to improve our real-time performance we should plan to run this algorithm on the computer's high-performance GPU to increase speed.
- Unimpressive stereo-vision range - In real-world testing, we found that our stereo-vision subsystem does not work very well if the objects are farther than 40 meters away. This is especially problematic for identifying pedestrians. In the future, we could work on adjusting the camera settings automatically depending on the type of environment and conditions detected by our system. This could improve performance at medium-range.

8. Project Management

The team's project management style has evolved continuously and significantly over the course of the project based on the effectiveness of certain techniques and the ineffectiveness of certain others. This section of the report contains a detailed evaluation of scheduling and task assignment approaches taken, the methodology behind the expenditure of the budget provided for the project, as well as an examination of the risk management and mitigation strategies used.

8.1 Schedule

The team faced multiple issues with meeting deadlines all the way from the start of the project to the very end due to a variety of reasons within the team's control and a few not explicitly under the team's control.

The major issue plaguing the team, specially at the start of the previous semester, was the lack of communication between team members. This led to widespread misunderstandings within the team and no clear sense of not only what a particular task entailed, but also to what extent one task affected the rest of the team's workflow. The establishment of a concise and succinct work breakdown structure seemed to be the solution to such problems. Contrary to the team's expectations, but reasonably so, the persisting issue was not caused by a lack of organization or clarity in how the tasks comprising the project were inter-dependent but due to a lack of consistency between members' understanding of exactly what each task meant. This was mainly caused by team members working independently on every task they were assigned, instead of collaborating and keeping checks on other team members.

The above issue wreaked more of a havoc in the second semester as the integration of subsystems is something that, by definition, cannot possibly be accomplished without the good faith effort of the team to work in a cohesive manner. This was evident with the fact that the team was unable to complete 100% of the goals for any of the progress reviews in the spring semester. Eventually, significant progress was made between the last progress review and the Spring Validation Experiments because of the team's collaboration and willingness to work with each other. This was further helped by a notable betterment in the team dynamics, which was perhaps caused by the last minute alignment of a common goal of showcasing a successful SVE demo. It was the introspection of each member's own contributions and the issues they faced that led the team to a stage where each teammate was willing to let go of their apprehension and skepticism of fellow teammates' competence and work towards a common goal.

Issues like these could have been averted had they been addressed in the initial stages of the project to create better interpersonal dynamics and a common understanding of what the project demanded of each teammate. This would have led to significantly more progress made by the team and would have avoided the stress that the team faced in final stages of the project.

8.2 Budget

The majority of the burden of the budget was alleviated by our sponsor, Delphi Automotive, providing the team with the two cameras, the lenses to go with them, and the radar unit. The majority of the team's budget was spent on the purchase of a high performance desktop computer with a state-of-the-art GPU, as planned. The remaining budget was spent in purchasing low-cost supporting material and devices like a step-up circuit PCB and the Adafruit Ultimate GPS. Table 7 lists the detailed breakdown of the expenditure by the team.

Table 7. Budget and expenditure breakdown

No.	Description	Cost	Quantity	Sponsored	Total
1	Grasshopper3 3.2 MP Color USB3 Vision (GS3-U3-32S4C-C)	\$ 975	2	Yes	\$0 (\$ 1950)
2	Delphi ESR 2.5 24V Radar	\$ 3300	1	Yes	\$0 (\$ 3300)
3	Tamron M118FM08, 8mm, 1/1.8", C mount Lens	\$ 210	2	Yes	\$0 (\$ 420)
4	Foneso 328ft Digital Handheld Rangefinder, Laser Distance Measurer	\$50	1	No	\$ 50
5	Belkin 6-Outlet Surge Protector	\$ 10	1	No	\$ 10
6	LM2587 Booster Converter Voltage Regulator (12V to 24V)	\$ 5	2	No	\$ 10
7	Adafruit Ultimate GPS Breakout	\$ 43	1	No	\$ 43
8	GPS external antenna and adapter cable	\$ 17	1	No	\$ 17
9	Kvaser CAN connector and adapter	\$ 380	1	No	\$ 380
10	Rectangle Glass 36" x 48"	\$ 108	1	No	\$ 108
11	CyberPower Uninterruptible Power Supply (UPS) 1350VA 815W	\$ 123	1	No	\$ 123
12	KRIEGER 1100W 12V Power Inverter	\$ 85	1	No	\$ 85
13	High Performance Desktop	\$ 3363	1	No	\$ 3363
Total budget left		\$ 5000 - \$ 4189 = \$ 811			

The budget was spent as planned for the intended SVE demo. Components like the 1100W Power Inverter and the 815W Uninterrupted Power Supply were not used due to the unexpected breakdown of the team's test vehicle, a Volvo S60, a few days before the SVE. This caused approximately \$200 of the spent budget to not be utilized. The other major failure was the purchase of the Kvaser CAN Adapter in the first semester. It was bought on recommendation from our sponsor, Delphi Automotive. In the second semester, Delphi suggested that the team should use the ethernet port instead of the CAN port from the ESR 2.5 to obtain detection-level data. This led to a loss of \$380 from the team's budget. Although, we do not believe that that purchase could have been avoided based on the trust we placed on our sponsor. The best return on investment was noted for the high-performance desktop computer as it became integral to the success of the project. It has a high potential, further beyond what is currently being utilized by the project, and would be beneficial in case of any future work performed on the project.

8.3 Risk Management

The risk management and mitigation approach taken by the team was not sufficient, as witnessed by the helplessness and panic caused by the sudden and unexpected breakdown of the project's test vehicle, the Volvo S60. This is not a risk that was predicted or planned for by the team. The team needed approximately \$800 to fix the issue that caused the breakdown, but the team had already spent the majority of the budget on other essential components. An emergency or a contingency budget of at least \$1000 should have been maintained to address the realization of such a risk.

Majority of the risks were identified and characterized accurately by the team, including, but not limited to, the inability to obtain data from the radar and the minimal dynamic range of the camera. Although, the steps taken to mitigate such risks were not enough or well thought out. One evident success of taking steps to mitigate risks was anticipating that the stereo-vision algorithm to compute the depth map could perhaps have a significantly high compute time. This risk was partially mitigated by down-sampling the images used to create the depth map. This was instrumental in achieving an acceptable frame rate for the depth map calculations. The next step in mitigating this risk was to parallelize the computation by using a GPU but that was not achieved in time for the final SVE demo. The project could have benefitted from more effective risk mitigation strategies.

9. Conclusions

Through the course of the project the team has learnt a lot and has gained vital technical competence and interpersonal experience. This section highlights some of the lessons that were learnt while working on this project and some suggestions for future work to further improve the project.

9.1 Lessons learnt

The following are some of the most important lessons that were learnt during the two semesters spent working on this project:

- For the success of any project a deep understanding between the teammates is essential and team collaboration is indispensable.
- Any experimental system is liable to breakdown no matter how well-tested. Thus, maintaining a contingency plan as well as keeping spare parts should not be overlooked.
- The probability of success of a project can be vastly improved by applying project management and systems engineering techniques.
- The work breakdown structure as well as planning the tasks should be dictated by the requirements.
- Unfiltered data for even an expensive sensor might look nonsensical but trying different filtering methods will almost always yield significantly better results.
- The integration of subsystems as well as 3rd party libraries may not always work as expected. So, allotting extra time for integration is a wise decision to make.
- Quantitative validation with proper metrics should always be preferred over qualitative judgment.

9.2 Future work

The following are avenues that might yield fruitful results if pursued in the future:

- Multiple Hypothesis Tracking by incorporating not only the position but also the velocity.
- Using an Unscented or Extended Kalman Filter with a polar coordinate system which is based on a constant acceleration model. In case of the radar, incorporating amplitude in the filter will significantly improve accuracy.
- Porting the code to use multi-threading in ROS instead of running on a single thread.
- The computation of the depth map should be parallelized to run on a GPU.
- Once high frame rates are achieved, implementing stereo-vision based tracking should show promising results.
- Radar can be trained to differentiate between vehicles, pedestrians and other obstacles with metallic content.
- Stereo Vision can be trained to mark drivable vs undrivable area.

10. **References**

- [1] Blackman, S. S. (2004). Multiple hypothesis tracking for multiple target tracking. IEEE Aerospace and Electronic Systems Magazine, 19(1), 5-18.
- [2] Blackman, S. S. (1986). Multiple-target tracking with radar applications. Dedham, MA, Artech House, Inc., 1986, 463 p.
- [3] Geiger, A., Roser, M., & Urtasun, R. (2010, November). Efficient large-scale stereo matching. In Asian conference on computer vision (pp. 25-38). Springer Berlin Heidelberg.
- [4] Dröppelmann, S., Hueting, M., Latour, S., & van der Veen, M. (2010). Stereo Vision using the OpenCV library.
- [5] Gehrig, S. K., Eberli, F., & Meyer, T. (2009, October). A real-time low-power stereo vision engine using semi-global matching. In International Conference on Computer Vision Systems (pp. 134-143). Springer Berlin Heidelberg.
- [6] Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C. Y., & Berg, A. C. (2016, October). SSD: Single shot multibox detector. In European Conference on Computer Vision (pp. 21-37). Springer International Publishing.
- [7] Hanselmann, T., & Morelande, M. (2007, July). Multiple target tracking with asynchronous bearings-only-measurements. In Information Fusion, 2007 10th International Conference on (pp. 1-8). IEEE.