Environment-Aware Sensor-Fusion for Traversable-Area Detection

Adrian Rechy Romero [*†], **Paulo Vinicius Koerich Borges** [*]
[*]Autonomous Systems Laboratory, CSIRO, Australia
[†]Autonomous Systems Lab, ETH Zurich

Abstract

We explore a *teach and repeat* approach using sensor-fusion for the detection of traversable areas, with application to autonomous vehicles. The method is suitable for vehicles which operate in a known environment, as is the case in many practical scenarios. By comparing a human-driven trajectory with the trajectory that would have been followed when using each sensor configuration, it is possible to automatically evaluate the performance of different sensors in different regions of the site. Thus, it is possible to define, for each region, what is the optimal sensor configuration as learnt from previous experience. Experiments are performed on a ground robot platform equipped with 2D laser sensors and a monocular camera with road detection capabilities. The results illustrate an increase in performance through a reduction of the failure rate along the trained route.

1 Introduction

Consider the industrial area depicted in Figure 1. The paths highlighted in green represent operation paths where an autonomous vehicle is to navigate on a daily basis to assist in a given task. To operate reliably in such an environment, most modern autonomous vehicle implementations use more than one sensing modality. Popular sensors include cameras, lasers, global positioning system (GPS), inertial measurement units (IMUs), among others. The use of multiple modalities is very effective where the reliability of sensors vary. This is the case in the different regions of Figure 1. From the image, it is clear that different parts of the site contain significantly distinct structural characteristics. The central path, for example, runs through a densely built up area, with buildings and large metal sheds. In contrast, other roads go through vegetation and unstructured areas, connecting the main buildings to other storage sheds and loading zones.

Figure 1: Satellite view illustrating a heterogeneous operation space. The scale is 410×360 meters.

In this challenging scenario, intelligent sensor fusion is essential to ensure dependable operation. In this paper we propose an informed sensor fusion strategy that considers knowledge of the environment, applicable to cases where the vehicle must operate in known areas.

By working under the assumption that the map is known, sensor performance can be mapped throughout the working area by analysing the behaviour of each sensor configuration on previous runs through the environment. Therefore, this approach chooses a sensor configuration based on their measured performance on practical tests without relying on rigid statistical models.

In many practical situations an autonomous vehicle is required to travel along a particular trajectory repeatedly. By manually driving the vehicle along specified

operation routes, it is possible to map its sensors' behaviours and understand where a particular sensor might show a low performance or even fail. By having a human driver demonstrating the desired response of the system it is also possible to stimulate the vehicle into mimicking the behaviour of the human driver. This can be achieved by selecting the configuration of inputs that produces outputs that better resemble those of the human operator. Therefore, by understanding the environment and its effect on the sensors, an autonomous vehicle can give different priorities to its inputs based on experience.

The proposed algorithm is evaluated on an test platform equipped with 2D lasers and a monocular camera with road detection capabilities. The vehicle is driven in an environment with different regions where the optimal sensor configuration changes. It is shown how, after analysing several manually driven runs, it is possible to automatically change the combination of sensors to improve the performance of the trajectory planning capabilities of the vehicle. Using as a metric of performance the resemblance to the human-driven trajectory.

1.1 Related work

The current paradigm in robotics involves a probabilistic modelling of the vehicle's state [Thrun, 2000]. Sensor fusion is then accomplished through the combination of information based on the probability distribution of each sensor's measurements. Different implementations of the Bayes Theorem such as *Bayesian networks* [Russell and Norvig, 2003], as well as linear approximations of them as the *Kalman Filter* [Maybeck, 1990], or Sequential Monte Carlo methods such as the *Particle Filter* [Doucet *et al.*, 2001; Thrun, 2002] are built upon the knowledge of the expected uncertainty of each sensor's information. This probabilistic models are difficult to obtain and may also be variable as different situations (e.g. different environments) are faced. This leads to situations where the system may be incorrect and yet confident about a certain measurement.

For the case of known trajectories and environments, various teach and repeat strategies have been developed for robot localization and control [Kidono *et al.*, 2002; Tang and Yuta, 2001; Ohno *et al.*, 1996]. Long-range rover autonomy has been achieved by using only a stereo camera pair [Furgale and Barfoot, 2010]. A similar method was used with an appearance-based LIDAR [McManus *et al.*, 2012] in order to obtain illumination invariance.

As for traversable-area detection, there is a large amount of literature available that describes the use of different sensors and algorithms that can detect roads and obstacles in either structured or unstructured environments (see [Bar Hillel *et al.*, 2014] for a comprehensive analysis of the state of the art).

This work presents a method for measuring the performance of a sensor fusion configuration for traversable-area detection based on the teach and repeat paradigm. This allows the system to tune the sensor configuration while trying to optimize the performance. A mapping can then be done from each region in the map to the optimal sensor modality.

2 Environment-aware sensor-fusion

This section describes the infusion of environment-awareness through sensor fusion.

2.1 Performance measure

The performance of a sensor configuration can be measured through the similarity between the behaviour of the vehicle while using that specific configuration and the behaviour followed when it is manually driven. In the case of traversable-area detection the system is expected to react to the obstacles around it in a similar fashion as a human driver. In this work we propose the use of a simulation of the vehicle's behaviour (*i.e.* an estimation of the trajectory that would have been followed) and a comparison with the manually driven trajectory to quantify the capabilities of each sensor configuration. A path planner that builds a trajectory based on the obstacles detected by the sensors could be used as such simulation.

Figure 2 illustrates the concept. While being manually-driven, the vehicle follows the red trajectory, a straight line. Meanwhile, for a specific sensor configuration the vehicle detects the obstacles represented by the shapes coloured in dark blue. At the beginning of the trajectory the vehicle detects the left-most obstacle, which is a false obstacle (*i.e.* the sensors incorrectly assume that there is something in front of the vehicle). The path planner would then generate a trajectory around it, depicted as a black arrow. The error at this point of the trajectory is high as there is a large difference between the trajectory followed and the one that would have been followed if that sensor configuration was to be used. As the vehicle moves forward, past the false obstacle, it reaches the middle point of the trajectory shown in 2. There are two obstacles at this point and the sensors correctly detect one of them but fail to detect the other (coloured in light blue). The path planner would then try to stay away from the seen obstacle without taking into account the presence of the unseen one. There is a small difference between the manually-driven trajectory and the one that the path planner generated, thus the error is reduced. Finally, at the rightmost section of the trajectory the sensors correctly detect the two obstacles and, trying to stay as far away from both obstacles as possible, the system plans a trajectory between them,

similar to the human driven one. As both trajectories are equivalent the error is reduced to zero.

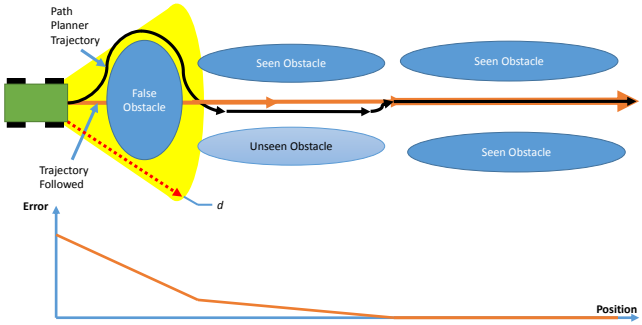


Figure 2: The error for a particular sensor configuration is equivalent to the difference between the driven trajectory and the trajectory that would have been followed when using that configuration. The red arrows depict the human-driven trajectory while the black arrows depict the path that the vehicle would follow while trying to avoid the obstacles that are detected by the sensors. The error is high at first, when the sensors incorrectly detect an obstacle. As it moves forward, the error decreases as the sensor readings become more correct. The yellow area is depicted to represent the points that lie within a distance d from the vehicle. The points in the trajectory that lie within this area are the ones that the system will analyse to measure the error at the current position.

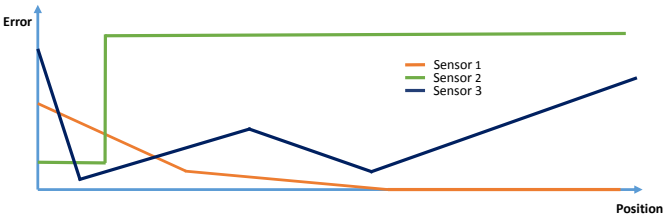
To evaluate a sensor configuration the trajectory is divided into overlapping sections. Thus, to analyse the effect of a particular sensor configuration at time t_1, the behaviour up to a time t_2, such that $t_2 > t_1$, is evaluated. Time t_2 can be defined through the euclidean distance d among the first $\mathbf{p}_d(t_1)$ and last $\mathbf{p}_d(t_2)$ points of the trajectory section. The points within distance d from the vehicle are depicted in Figure 2 as a yellow area.

$$|\mathbf{p}_d(t_2) - \mathbf{p}_d(t_1)|^2 = d. \tag{1}$$

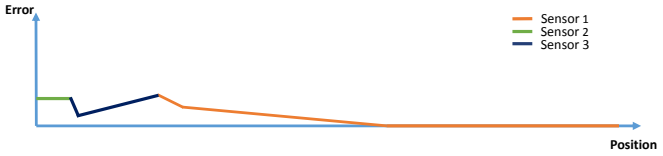
With the first and last poses defined it is possible to command the corresponding path planner to generate a trajectory $\mathbf{p}_p(t, \lambda)$ based on the input of the current sensor configuration. The error J_λ for sensor configuration λ at time t_1 is defined as the integral of the squared difference between the two trajectories.

$$J_\lambda(t_1) = \int_{t_1}^{t_2} |\mathbf{p}_d(t) - \mathbf{p}_p(t, \lambda)|^2 dt \tag{2}$$

Once the performance of every sensor configuration has been evaluated for a particular position, the optimal sensor configuration $\lambda_{opt}(t_1)$ can be defined as the one that yielded the lowest error. This is illustrated on Figure 3.



(a) Performance graph of three sensors. Their performance varies with the position.



(b) The objective is to select the sensor configuration that yields the lowest error on each position.

Figure 3: Performance is improved by selecting the sensor configuration that yielded the lowest error on previous runs.

$$\lambda_{opt}(t_1) = \arg\min_{\lambda} J_\lambda(t_1) \tag{3}$$

The optimal sensor configuration for a position $\mathbf{p}_{current}$ can be approximated by the sensor configuration from the point in the trajectory that is closest to it.

$$t_{closest} = \arg\min_{t} |\mathbf{p}_{current} - \mathbf{p}_d(t)|^2 \tag{4}$$

$$\lambda_{opt}(\mathbf{p}_{current}) \approx \lambda_{opt}(t_{closest}) \tag{5}$$

The training process can be performed offline, by first recording a manually-driven trajectory with all the sensor readings and then analysing each sensor configuration through a simulation. At runtime, lookup methods can be used to save computational power and memory. For instance, the sensor configurations can be mapped into the different positions along the map through Equations (4) and (5).

3 Experimental Set-up

3.1 Practical Considerations

The algorithm is implemented in `C++` and runs on the Robotics Operating System (ROS) [Quigley *et al.*, 2009]. For the lasers, four Hokuyo range sensors were used. The sensors were placed on the left right corners, in the front and back of the vehicle, both mounted at height $h_{\text{laser}} \approx 0.79$ m. The laser scanners have an angular resolution of 0.25° and a scan angle of 270°. The laser scanners were running at a frequency of 30 Hz and have a maximum detection distance of 30 m. The vision sensor was composed of a colour Point Grey Chameleon camera holding a 1/3" CCD imaging sensor fitted with Kowa lenses of 5 mm focal length and aperture $F1.8$. The resolution was set to 800×600 pixels running at a frame rate of 15 Hz. The system was mounted at a height of 1.0 m. The sensors were mounted on a John Deere Gator, an electric medium size utility vehicle (see Figure 4), and driven manually and autonomously in the unstructured and structured industrial environment shown in Figure 5.

Figure 4: The test vehicle, a John Deere Gator holding multiple sensors (lasers, camera, wheel odometry) was used.

As seen in this figure, some sections of the road are delimited by low grass and bushes which change frequently and are weak features for the 2-D laser sensors. Similarly, some sections contain buildings with similar colour as the road and a tendency to create strong shadows that affect the estimation of the visual road detection algorithm. Through the sensor-fusion method presented, we expect the system to recognize which sensor configuration yields the best performance on each region. Some effects of the environment change with time, as do the shadows; however, they are still region-bounded as it is the building structure that tends to create such shadows. Robustness can be achieved by analysing several runs (at different times of the day and illumination conditions) and avoiding the use of sensors in areas where they have shown weak performance.

Among the relevant packages from ROS, we employ the *Navigation stack* [Marder-Eppstein *et al.*, 2010] and, more specifically, the *costmap_2d*, and *move_base* packages. The former provides a framework to combine the sensor information into a two-dimensional costmap that is used by the latter to generate trajectories for the vehicle to follow.

3.2 Visual road detection

The visual road detection algorithm is based on an Artificial Neural Network. It works in a pixel-wise manner with the following inputs.

- The three *RGB* channels.
- An illumination invariant image as described in[Alvarez and Lopez, 2011].
- A gradient image created by applying a Scharr filter followed by morphological operations. This created an entropy filter that was useful for detecting the vegetation.
- The pixel coordinates.

The images used as inputs are illustrated on Figure 6.

Labelling of the training data was performed automatically as seen in Figure 7. The area inside the red rectangle was always assumed to be background while the area inside the blue rectangle was always assumed to be road. This is a similar approach as used in CITE. Furthermore, the neural network that was trained with this data was then used to automatically label a second set of training data. This allowed the introduction of the pixel coordinates as input parameters. Figure 8 shows the improvement.

3.3 Cost-map fusion

This section defines the sensor fusion method used for the test platform. The idea is to map the obstacles from every sensor into a costmap, giving different weights to each sensor depending on the level of reliability expected.

For the case of n different sensors the combined costmap $\mathbf{C}_t \in \mathbb{R}^2$ is obtained by combining the costmaps generated by each sensor $\mathbf{C}_1, \mathbf{C}_2, \ldots, \mathbf{C}_n$, as follows.

$$\mathbf{C}_t = \alpha_1 \mathbf{C}_1 + \alpha_2 \mathbf{C}_2 + \ldots + \alpha_n \mathbf{C}_n, \tag{6}$$

$$\lambda = [\alpha_1, \alpha_2, \ldots, \alpha_h]^T \in \mathbb{R}^{n \times 1}. \tag{7}$$

Where α_i represents the weight assigned to the costmap $\mathbf{C}_i$ and λ corresponds to a particular sensor configuration, that is, a specific collection of weights for the sensors.

Two sensor configurations were tested where the sensors' costmaps were either enabled or disabled. This

means that the α_i parameters were given binary values. This was required due to the structure of the *costmap_2d* package. In the case of the test platform used in this experiment three costmaps were used. The first one corresponded to the map of the environment, which was always enabled. The other two correspond to the obstacles detected by the 2d lasers and the obstacles detected by the visual road detection algorithm.

The *costmap_2d* package inflates the obstacles, assigning the highest cost to the position where the obstacle is located and a decreasing cost around it as the distance from the obstacle increases. Figure 9 shows a visualization of the costmap while the vehicle is moving through it. The dark-blue points are the obstacles as seen by the visual road detection algorithm while the red squares correspond to the obstacles detected by the lasers. The combined costmap is represented by the different tonalities in the map. They go from light yellow (highest cost), to cyan, pink and purple (lowest cost) where each colour represents a different cost. The grey areas in the map correspond to free-space (no cost associated). Figure 9 also illustrates how the sensor configuration changes along the trajectory. At the left side of the image the costmap built with the visual road detection was not being used and the combined costmap was built only with the map and laser costmaps. Then, the costmap that corresponds to the visual road detection was enabled, which found obstacles that were closer to the vehicle.

3.4 Practical Considerations

The framework used set up some constrains for the experiment

- Based on the range of the sensors used, the trajectory was divided into sets of $d = 20$ meters.
- The human-driven trajectory was built with the output of a laser localization algorithm.
- Only two sensor configurations were tested: *Lasers* and *Lasers with Vision*. The vision based road detection was not tested by itself since the lasers are required as a security measure.
- Since the camera sensor's behaviour is also a function of the time of the day, several runs where analysed, where a grid map of the performance of each sensor configuration was created on every lap. The results were then merged in order to obtain a single grid map for every sensor configuration. The merging was performed by averaging the grid maps in a cell-wise manner.
- In total 6 training laps and 6 testing laps were performed
- Due to the structure of the *costmap_2d* package, the weights of the sensor configuration λ were assigned binary values. Thus, each configuration was a combination of enabled and disabled sensors.

Additionally, since the system gives both the driven trajectory $\mathbf{p}_d$ and the path planner's trajectory $\mathbf{p}_p$ as a collection of discrete poses, Equation (2) was adapted as follows.

$$J_\lambda(k_1) = \frac{1}{k_2 - k_1} \sum_{k_d=k_1}^{k_2} \min_{k_p} |\mathbf{p}_d(k_d) - \mathbf{p}_p(k_p, \lambda)|^2 \quad (8)$$

Where k_1 and k_2 are the discrete equivalents of t_1 and t_2. A distinction is made among the samples of the driven trajectory k_d and those of the trajectory generated by the path planner k_p since they do not always contain the same amount of discrete positions. Similarly, the cost is averaged as to not distinguish among trajectories formed by a different amount of discrete positions. Thus, the cost represents the average distance between the points in the driven trajectory and the points closest to them from the trajectory built by the path planner.

In order to quantify the robustness of the system the concept of high error measurements is introduced, where a high error measurement corresponds to cases where the distance between the two trajectories is considered too large for the vehicle to navigate safely. In this case an average distance τ meters between both trajectories, as defined in Equation (8) is used. Thus, the percentage of high error measurements per trajectory can be defined as follows.

$$\text{PHEM} = \frac{1}{k_{end} - k_{start}} \sum_{k_{start}}^{k_{end}} [J_\lambda(k) \geq \tau]. \quad (9)$$

Where k_{start} and k_{end} correspond to the beginning and end of a trajectory respectively and the notation $[J_\lambda(k) \geq \tau]$ corresponds to the Iverson notation where it denotes a value of 1 if the condition is satisfied and 0 if not. Thus, Equation (9) corresponds to getting the percentage of cases per trajectory where the error is greater than τ.

4 Results

Table 1 shows the percentage of trajectory sections that showed a high error via the PHEM as defined in Equation (9). Similarly, Figure 10 shows a graphical representation of the PHEM with the different sensor configurations. There is a reduction in the Percentage of High Error Measurements when the *Environment-Aware Sensor-Fusion* is introduced. As a reference, a full lap around the trajectory includes approximately 1000 measurements, and the width of the road at its narrowest point measures approximately 3 meters. If a deviation of 3 meters from the trajectory is considered unreliable,

Table 1: The percentage of trajectory sections that yielded a high error is shown for each sensor configuration and for the environment-aware sensor-fusion method. High error measurements were defined as those that have an average distance between trajectories of more than τ meters. They represent the situations where the sensors incorrectly detected an obstacle, causing a behaviour in the vehicle that did not resemble the one followed when it was manually-driven (As a reference, the width of the road at its narrowest point is of approximately 3 meters). The environment-aware sensor fusion method showed a decrease in the high error rate, which translates into more robustness.

	PHEM		
Sensors conf.	$\tau = 3$	$\tau = 2$	$= 1$
Lasers only	0.0050	0.015	0.0378
Lasers and Vision	0.1655	0.1752	0.2214
E.A. Sensor-fusion	0.0018	0.0033	0.0163

the *Environment-Aware Sensor-Fusion* would only reach this value in 1.8 trajectory sections, against 5 and 16.5 occasions for the *Lasers* and *Lasers with Vision* combination. Thus, the *Environment-Aware Sensor-Fusion* method has a reduction of 62.5% when compared to the *Lasers* and a reduction of 98.7% when compared to the *Laser and Vision* combination.

The performance of the two sensor configurations has been plotted for the six training runs in Figure 11a. The higher error values are those where the sensors incorrectly detected an obstacle and the path planner had to look for a long trajectory around it or alternatively failed to find one. These were clipped at a value of 3 for visualization purposes. This occurred frequently when the visual-based road detection algorithm incorrectly detected an obstacle due to the strong shadows produced by the buildings or the trees. Figure 11b shows the distribution of errors along the map. It is evident, from observing the repeatable behaviours among the different runs, that there is a correlation between the map region and the sensor performance.

A mapping from position to optimal sensor configurations was built through equations 4 and 5. Figure 12 shows the optimal configuration along the trajectory.

The results of using the new configuration map on the testing runs is shown in Figure 13. By comparing the error plots to the ones from Figure 11, it is evident that there was a decrease in the amount of high-error readings.

5 Conclusion

This work presents an approach for obtaining environment awareness in sensor fusion. It is shown that in environments where the sensor behaviour changes with the region in the map, it is possible to select an adequate sensor configuration based on experience from previous runs. Experimental results show a higher robustness as measured by the amount of high error readings.

Future work could use a similar methodology for other applications in navigation, such as localization, where the sensor behaviour also varies along the different regions in the map.

References

[Alvarez and Lopez, 2011] J.M.A. Alvarez and A.M. Lopez. Road detection based on illuminant invariance. *Intelligent Transportation Systems, IEEE Transactions on*, 12(1):184–193, March 2011.

[Bar Hillel *et al.*, 2014] Aharon Bar Hillel, Ronen Lerner, Dan Levi, and Guy Raz. Recent progress in road and lane detection: a survey. *Machine Vision and Applications*, 25(3):727–745, 2014.

[Doucet *et al.*, 2001] Arnaud Doucet, Nando De Freitas, and Neil Gordon. *Sequential monte carlo methods in practice*. Springer-Verlag, 2001.

[Furgale and Barfoot, 2010] Paul Furgale and Timothy D. Barfoot. Visual teach and repeat for long-range rover autonomy. *Journal of Field Robotics*, 27(5):534–560, 2010.

[Kidono *et al.*, 2002] Kiyosumi Kidono, Jun Miura, and Yoshiaki Shirai. Autonomous visual navigation of a mobile robot using a human-guided experience. *Robotics and Autonomous Systems*, 40(23):121 – 130, 2002. Intelligent Autonomous Systems - {IAS} -6.

[Marder-Eppstein *et al.*, 2010] Eitan Marder-Eppstein, Eric Berger, Tully Foote, Brian Gerkey, and Kurt Konolige. The office marathon: Robust navigation in an indoor office environment. In *International Conference on Robotics and Automation*, 2010.

[Maybeck, 1990] PeterS. Maybeck. The kalman filter: An introduction to concepts. In IngemarJ. Cox and GordonT. Wilfong, editors, *Autonomous Robot Vehicles*, pages 194–204. Springer New York, 1990.

[McManus *et al.*, 2012] C. McManus, P. Furgale, B. Stenning, and T.D. Barfoot. Visual teach and repeat using appearance-based lidar. In *Robotics and Automation (ICRA), 2012 IEEE International Conference on*, pages 389–396, May 2012.

[Ohno *et al.*, 1996] T. Ohno, A. Ohya, and S. Yuta. Autonomous navigation for mobile robots referring pre-recorded image sequence. In *Intelligent Robots and Systems '96, IROS 96, Proceedings of the 1996 IEEE/RSJ International Conference on*, volume 2, pages 672–679 vol.2, Nov 1996.

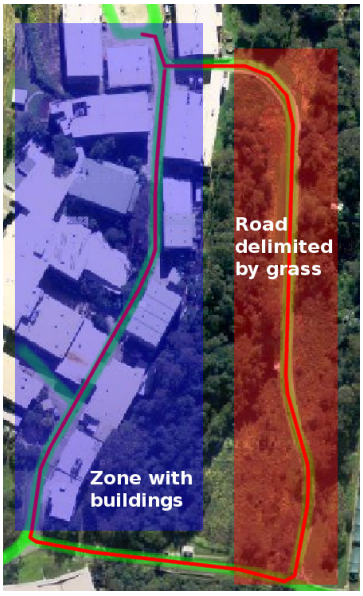
[Quigley *et al.*, 2009] Morgan Quigley, Ken Conley, Brian P. Gerkey, Josh Faust, Tully Foote, Jeremy Leibs, Rob Wheeler, and Andrew Y. Ng. Ros: an open-source robot operating system. In *ICRA Workshop on Open Source Software*, 2009.

[Russell and Norvig, 2003] Stuart J. Russell and Peter Norvig. *Artificial Intelligence: A Modern Approach*. Pearson Education, 2 edition, 2003.

[Tang and Yuta, 2001] Lixin Tang and S. Yuta. Vision based navigation for mobile robots in indoor environment by teaching and playing-back scheme. In *Robotics and Automation, 2001. Proceedings 2001 ICRA. IEEE International Conference on*, volume 3, pages 3072–3077 vol.3, 2001.

[Thrun, 2000] S. Thrun. Probabilistic algorithms in robotics. *AI Magazine*, 21(4):93–109, 2000.

[Thrun, 2002] Sebastian Thrun. Particle filters in robotics. In *in Proceedings of the 17th Annual Conference on Uncertainty in AI (UAI*, 2002.



(a) The trajectory to be followed is marked in red. Similarly, different regions where the sensors may not perform correctly are highlighted.

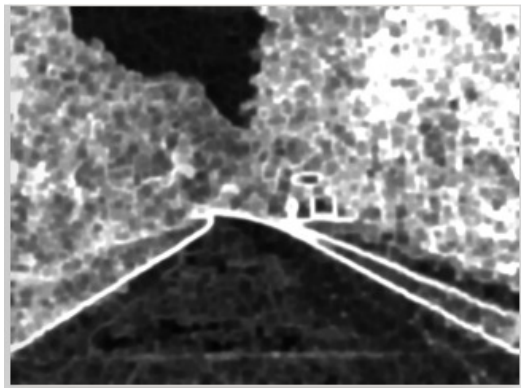
(b) This region has a road delimited by grass. A laser set-up can not detect the grass as a non-traversable area.

(c) The colour of the sidewalk and buildings in this section may cause false positives in the visual road-detection algorithm. Similarly, the strong shadows also affect the outcome of the algorithm.

Figure 5: The sensors' performance varies along the different sections of the trajectory

(a) RGB channels.

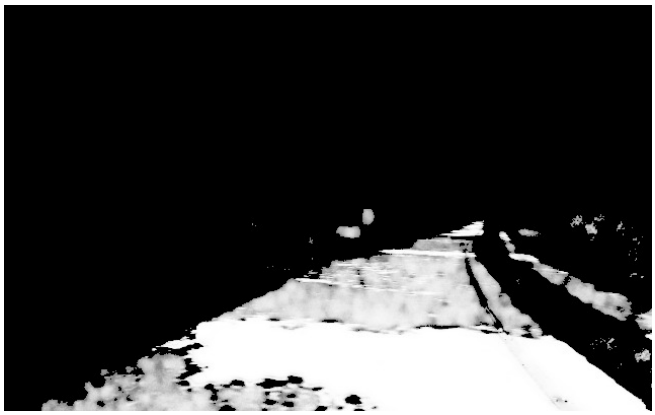
(b) Illumination invariant image. Notice the reduction of shadows.



(c) Gradient image built with a Scharr filter followed by morphological operations.

Figure 6: The input parameters for the neural network are shown. Additionally, the pixel coordinates were also used.

Figure 7: This image shows the auto-labelling procedure. Pixels taken from the red rectangle were labelled as background while pixels taken from the blue rectangle were labelled as road.



(a) Neural network trained with the labels created from the rectangles in Figure 7. The road is not consistent.

(b) By using the first neural network trained to label the data, the pixel coordinates could be included as parameters, which improved the performance of the classifier.

Figure 8: The two steps of the training process for the visual road detection algorithm. A first neural network was used to label the data for a second neural network.

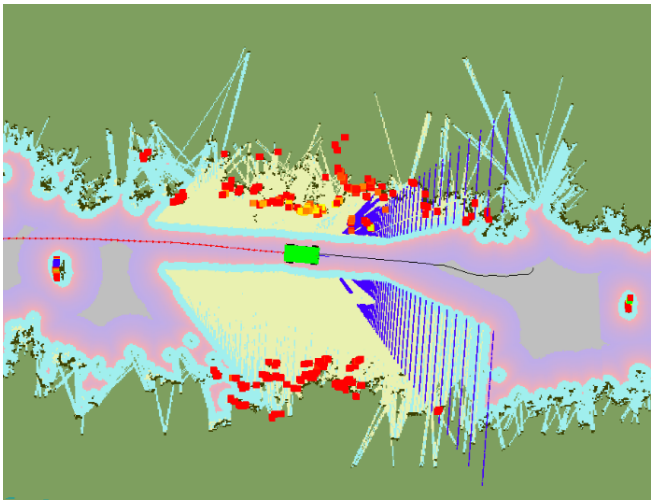


Figure 9: Visualization of the vehicle traversing through the costmap. The blue lines represent the obstacles as detected by the visual road detector while the red squares are the obstacles detected by the laser. A change of the sensor configuration, first without vision and then both sensors, can also be seen.

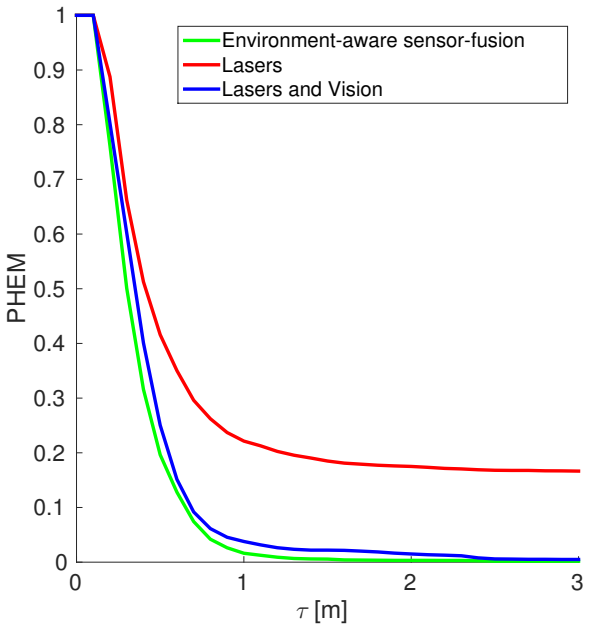
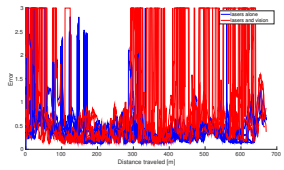
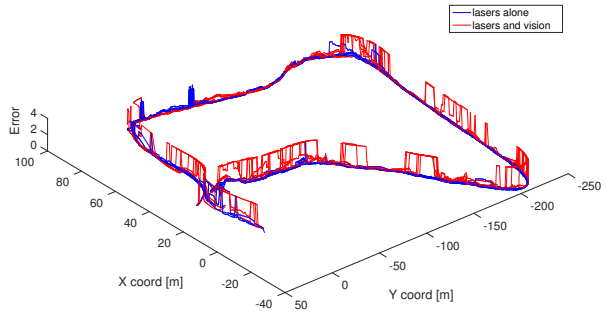


Figure 10: Graph that shows the percentage of high error measurements (PHEM) vs the average distance among both trajectories τ as described on Equation (9)



(a) Performance graph of the sensors in 2D.



(b) Performance graph of the sensors in 3D. Due to the similar behaviours among different runs it is evident that there is a correlation between the region and the sensor performance.

Figure 11: The error incurred by each sensor is plotted among different runs. The error is clipped at a value of 3 to facilitate visualization. The sections of the trajectory that have such a high error correspond to the cases when the sensors incorrectly detected an obstacle, making the path planner build a much larger trajectory than needed.

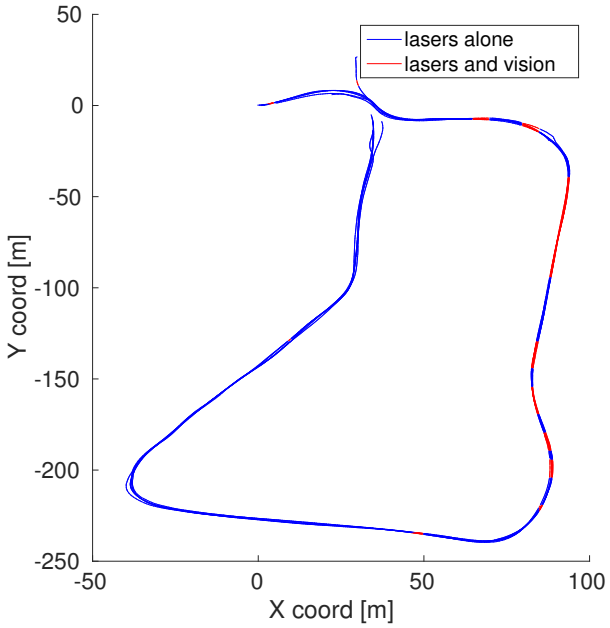
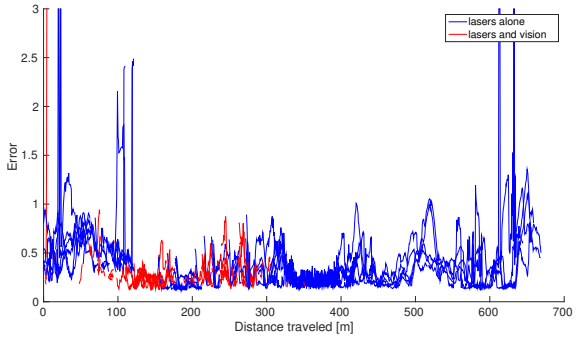
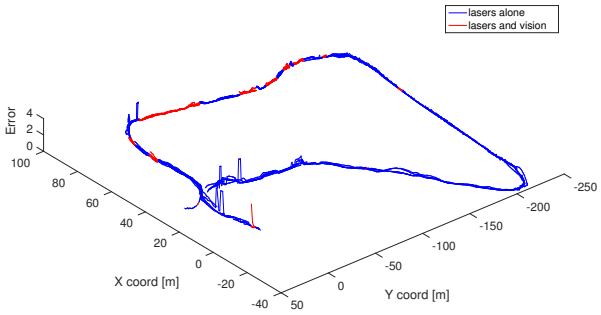


Figure 12: Map of the trajectory that shows the optimal sensor configurations along the route



(a) Performance graph in 2D.



(b) Performance graph in 3D.

Figure 13: The error incurred by the optimized sensor configuration is plotted among six runs. The error remained low among most of the trajectory. The error is clipped at a value of 3 to facilitate visualization.