Evaluation of Effective Sensor Positioning in Autonomous Vehicles via Simulation

Abstract— **Autonomous vehicles have the ability to plan, map, navigate, as well as avoid static or dynamic obstacles when exploring an environment. In this paper we show how the number of sensors can affect the power usage in the car, and how we can get an optimal set of sensors to perform autonomous driving efficiently. We use real world data of a small-scale car to gather electrical power usages, along with a simulation over several obstacle courses, to show that sensor placement can significantly alter the total energy used by an autonomous vehicle. Our results show that no one specific sensor configuration fares best in all scenarios, however, they demonstrate a general pattern that sensors in the direction of travel grant greater reductions in energy usage. Furthermore, our results show that adding sensors opposite to the direction of travel tends to only increase energy usage. We also found that there is a scenario where adding sensors does not decrease the time to reach the target position but only increases the energy usage.**

Keywords— autonomous vehicles, energy usage, sensor positioning

I. INTRODUCTION

Autonomous vehicles rely on sensors to measure road condition and make driving decisions, and their safety relies heavily on the reliability of the sensors. Out of all obstacle detection sensors, ultrasonic sensors have the largest market share and are expected to be increasingly installed in autonomous vehicles [5]. These sensors are able to detect obstacles by emitting ultrasounds and analyzing their reflections. An ultrasonic sensor is an ultrasonic transducer which converts high frequency ultrasound, detected after reflection from obstacles, into electrical signals [1].

The purpose of this work is to help design a sustainable autonomous vehicle by analyzing the power consumption based on the sensor positioning and observe which improvement can be done for the sensor placement and how many sensors would be enough to provide us with an efficient result.

In this work we experiment with different sensor configurations on an autonomous car in order to recognize how the configuration affects energy usage. In our case autonomous car navigates from a starting position to a goal position using ultrasonic sensors as aid to detect the physical environment. We worked on different adjustments in the sensor configurations and number of sensors used on the car's chassis and try to see how the electrical energy consumption depends on the ultrasonic sensor configurations. Our goal is to determine an optimal number of sensors and a configuration to be used to reach the goal position with a minimal amount of energy. We assume that after a certain number of sensors have been installed, there will be no benefit of adding new sensors with regards to the time it takes to reach the target position.

The structure of the paper is as follows. The next Section presents some background knowledge and Section III briefly overviews related work. Section IV describes methods used in this work and Section V considers obtained results. Section VI includes discussion, and the final Section contains conclusions and future work.

II. BACKGROUND KNOWLEDGE

In this Section we briefly introduce the main concepts related to sensors used for positioning of autonomous cars.

A. *Ultrasonic sensors*

The Ultrasonic Sensor module sends out the 8 bursts of 40kHz ultrasound and listens to the echo [2]. The time taken by the echo to reach the receiver is used to determine the distance between the sensor and the obstacle from which it got reflected. It is simply based on the speed formula, as we know the speed of the ultrasound fired from the sensor's transmitter.

B. *Mapping*

Grid based maps are considerably easy to learn, because they facilitate accurate localization, and they are easy to maintain. This approach uses a simple identifier for presenting a specific landform in each cell of an overlain grid [9]. Grid based mapping is typically used when we deal with a small size environment, whereas topological mapping is used when we deal with an exceptionally large environment. The sonar sensor measures the appropriate distances to the nearby obstacles, along with the noise. To build the metric maps, the sensor reading is translated into the occupancy grid for each grid cell. Since, we are dealing with a $1/10^{th}$ scale of an actual vehicle size, we have a very small-scale environment. This is why we decided to go for grid-based mapping to simulate the environment and observe how it would affect the electrical parameter usage when we run the car with N-sensors and at different configurations. In grid-based maps, each grid cell **<*x, y*>** in the map has an occupancy value attached to it, which measures the robot's subjective belief whether or not its center can be moved to the center of that cell [8]. When considering the entire scenario, some of the constraints need to be taken into account, e.g., not colliding with an obstacle in an environment is a hard constraint, and heading velocity, turning angle will be the soft constraints.

C. A* Algorithm

We use A* search algorithm as the path finding algorithm. It is an extension of Dijkstra's algorithm [14] that tries to reduce the total number of states explored, by incorporating a heuristic estimate of the cost to get to the goal from a given state. In this algorithm we use the sum of the functions for the cost-to-come from state X to state X_I, and the estimate for the cost-to-go from state X_I to X_g given as $C^*(x) + G^*(x)$, implying that the priority queue is sorted by the estimates of the cost-to-go from X_I to X_g. Also, when the cost-to-go is underestimate of the true optimal cost-to-go, then we have a guarantee that the **A*** will find the optimal path [10].

III. Related Work

The research in the field of our work has been mostly focused on the different sensors used in autonomous vehicles and their comparison based on the performance. [9] shows the use of grid-based mapping approach for providing an efficient solution to the problems of mapping small landforms over large areas by providing a consistent and standardized approach to spatial data collection. [7] models environment using a set of polynomials as SLAM landmarks along with EKF for efficient mapping of the environment. [3] discusses some of popular sensors used in autonomous vehicles and their technical specification with regards to link range, power used etc. and the types of attacks these sensors are prone to.

[12] and [8] give clear cut insights into the theoretical and mathematical background for mapping an environment. Both authors explain how mapping algorithms estimate the posterior probability for individual grid cells and how the fusion of the data into the map from multiple sensors takes place.

However, we found that the field lacks dedicated research that looks directly at the electrical power usage and optimal sensor positioning for autonomous vehicles.

IV. Methods

This section describes two methods. The first one is a simulation method where we simulate a car traversing through a predefined static map with obstacles. We check how the sensor positioning affects the autonomous driving of the car, how sensor activation takes place and what amount of energy the car consumes. In the second method we conducted a practical experiment to measure the power drawn from a real autonomous car and the ultrasonic sensor. The data from this experiment were then fed into the simulation to see how much power would have been consumed in different configurations. The code was written in Kotlin[1] due to our familiarity with the language. Source code and data files for the project are available via a *git* repository[2].

A. Method 1: Simulating an autonomous car

The simulation considers a robot agent in a grid world. The grid world is represented by a $<N \times M>$ Boolean grid, where true values represent obstacles in the world. The grid world follows Moore's grid world rules [13], where every cell has 8 neighbors and diagonal traversal is allowed. The grid world of obstacles is called the "real world" in order to distinguish it from the robot's "known world". The robot's "known world" is a second $<N \times M>$ grid which contains all obstacles picked up by the robot's sensors.

The robot was represented by a point in the grid, with an orientation, a set of sensors, and the "known world" grid. The robot always starts at the bottom left of the grid and traverse the world until it reaches the top right cell of the world. We defined a direction enumeration to represent the neighbors of a cell as North (N), North-East (NE), East (E), South-East (SE), South (S), South-West (SW), West (W), and North-West (NW). The robot's sensors are treated as pointing to one of these directions relative to the robot and they are fixed to the robot and rotates when the robot turns. This means that, when the robot is faced North its North sensor faces North and its North-West sensor faces North-West and if the robot turns to the East, its North sensor faces East and its North-West sensor faces North-East. With this setup the robot could have a maximum of 8 sensors. The robot is always given a North facing sensor. This is done to prevent the robot from driving into any obstacles. All sensors had the same range for detecting obstacles.

The sensors are implemented in a way that when activated they scan along their direction relative to the robot until an obstacle is found and then update the robot's known world. For example, if a sensor facing North relative to the robot's orientation then it looks at each successive cell in the robot's x column until an obstacle is found, up to a range of 4 cells away. To move to a new cell, the robot first calculates how many turns were necessary to face the new cell. If the robot turns to face each cell along the turn to the new cell and triggers its sensors, then it triggers sensors when facing the new cell. At the end of turning, the robot re-calculates its trajectory if it's known world had been updated by the sensors. Then it picks the next cell to move to according to the new path and repeat this process until it moved to the next cell. With this procedure, the robot would not walk into obstacles in front of it and optimize its path as soon as new information was gathered. The A* algorithm implemented for the robot recalculates the path whenever the robot's known world was changed. The heuristic used for the A* algorithm was "diagonal distance", as the robot moved from grid cell to grid cell using Queen rules in chess. The diagonal distance heuristic is calculated as the greater value of either the absolute value of the difference between the goal's X coordinate and the robot's current X coordinate, or as the absolute value of the difference between the goal's Y coordinate and the robot's Y coordinate. Five obstacle maps were created according to the specifications above. The examples of maps are given in the Results Section. Each map was created with at least one possible path from start to goal.

For each map, 10 robots with different sensor configurations traversed the map according to the algorithm described above. One of them is a control robot that only had an N sensor. This robot serves for the minimum comparison

---

[1] https://kotlinlang.org
[2] https://github.com/LedanDark/Research_Simulation

to the other robots. The sensor configurations chosen were predicted to fare well, one with sensors on all directions, and semi-randomised configurations. Each robot traversed their own version of the obstacle course. Each robot then reported how many cells it traversed, how many turns it executed, and how many times its sensors were activated.

B. Method 2: Gathering power usage data from small scale car.

An autonomous car was used to obtain real world data. The obtained data are used as an approximation for the different costs the simulated car would incur while performing different moves. We gathered the voltage (V) readings from the car for moving forward, backward, turning right or left. The same gathering is applied for current (A) reading. Then we calculated the power (Watt) consumption for every move the car executed. The formula used to calculate the power consumed is given below.

$$\boldsymbol{P = V \times I}$$

The difference in the simulation and the real-world experiment is as follow. The simulation considered that the car takes one step at a time to move from one grid cell to another like a discrete value, but in the real-world situation the car moves continuously in the environment. So, we measure the turning radius of the real car and infer the area of the circle covered by the real car for turning moves to the size of the grid cell in the simulation. In this way we had a common ground where the characteristic behavior of the real car and the simulated behavior for moving from starting point to end point is used for analysis and comparison.

V. Results

The cost of turning is based on the average power consumed by the real car for turning right/left when moving forward as well as backward (see Table I). The cost of turning left was found to be higher than the cost of turning right. Inspection of the turning motor of the car would interfere with the driveshaft of the car when turning left by pressing down on the driveshaft. The driveshaft would be unimpeded by the turning motor when turning right. For this work the unimpeded turning cost was used.

TABLE I. Energy Usage of Car Actions at Different Speed values which are inputed to car from a program which drives the car at different speed, the speed in the table is not the actual speed but the value inputed using W-A-S-D keys to drive the car. [https://github.com/ECC-BFMC/BFMC_Startup.git]

Cost of turning (Watt)	Cost of sensor (Watt)	Cost of traversing one cell (Watt)	W-A-S-D Value for Speed
2.61±0.4	0.03±0.008	2.53±0.15	15
4.62±0.4	0.03±0.008	4.18±±0.15	19
6.28±0.4	0.03±0.008	6.39±±0.15	25
9.1±0.4	0.03±0.008	8.98±0.15	30

The cost of using the turning and traversing one cell is found to be roughly equivalent. The cost of using the sensor was found to be marginal when compared to the cost of traversing or turning.

As stated in Section IV, different maps and sensor configurations were used to collect data. Map-3 is the simplest map with a chess style layout of obstacles and free space. Fig. 1 shows the map for scenario 3 in graphical form, where free space appears as grey squares and obstacles space appears as dark grey squares. The green squares represent the path that was taken by the simulated robots. Table II shows the energy used to traverse this scenario for different sensor configurations, where the estimated cost was calculated using the simulation moves multiplied by the approximated cost of each move.

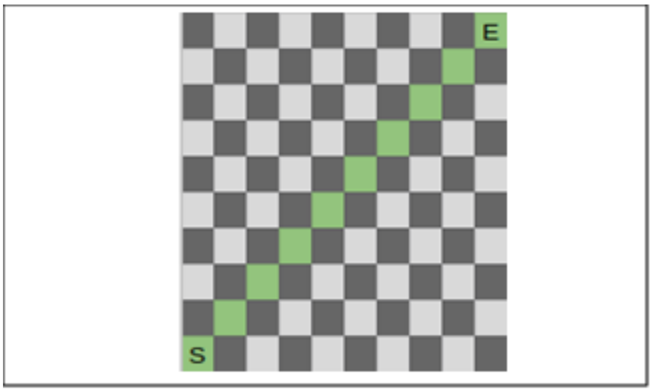

Fig. 1. Traversal of simulated car with North Sensor over Map 3

TABLE II. Map 3 Sensor Configuration Energy Usage

Sensor Configuration	Turns	Sensor usages	Dis-tance	Potential Efficiency Ratio	Esti-mated cost (W)
N _	1	10	9	NA	25.68
N _ SE _ SW _	5	42	9	-0.125	37.08
N _ NE _ NW _	3	36	9	-0.077	31.68
N _ NE _ S _ NW _	3	48	9	-0.053	32.04
N _ NE _ NW _ SE _ SW _	3	60	9	-0.040	32.4
N _ NE _ E _ NW _ W _	3	60	9	-0.040	32.4
N _ NE _ E _ SE _ S _ SW _ W _ NW _	3	96	9	-0.023	33.48
N _ S _	1	20	9	0.000	25.98
N _ S _ E _ W _	1	40	9	0.000	26.58
N _ E _ W _	1	30	9	0.000	26.28

In scenario 3 all robots took the same path but used different numbers of turns. As can be seen in Table II, in this scenario there was no sensor configuration that could be found that would use less energy for the traversal than the simplest configuration N.

In Map 2, the layout is more complex. Fig. 2 shows the traversal of a car with all 8 sensors attached. In this figure, blue color denotes the obstacles detected by the autonomous car during its run. Table III shows the energy usage of different sensor configurations for their traversal over this map.

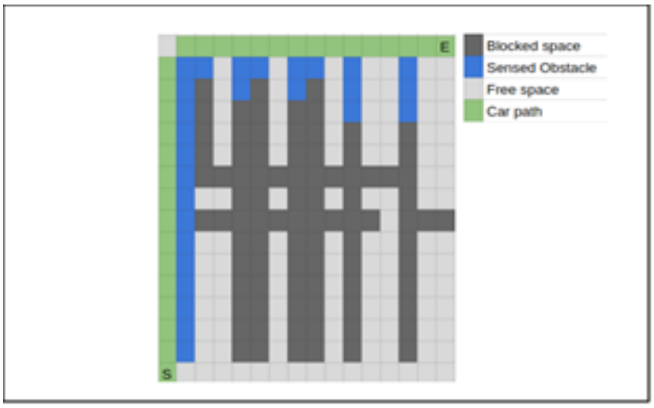

Fig. 2. Traversal of simulated car with 8 sensors over Map 2

In this scenario, the configuration that had only south facing sensors (N&S, and N&SE&SW) performed worse than the car with only an N sensor. These configurations took more than 29 cells to traverse the map, whereas all other configurations took 29 cells to traverse the map. Sensor configurations that included N, NE, and NW sensors reduced the estimated energy consumption from ~200 Watts to ~100 Watts. This is a reduction of more than 50%, when compared to the configuration with only a N sensor. In this set of sensors with more than 50% energy reduction, adding more sensors further reduced the energy consumption from ~100 Watts to ~91 Watts. The configuration with least energy usage was the car with all 8 sensors attached. The highest allowable energy cost for the sensors was ~1.533, from the configuration with N, NE, and NW sensors.

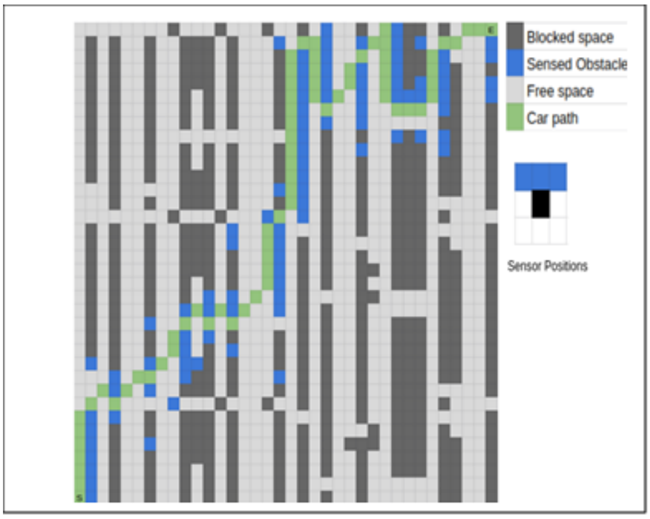

Fig. 3. Traversal of simulated car with 8 sensors over Map 1

Map-1, shown in Fig. 3, was the largest map generated for this project. In this scenario N&S and N&SE&SW sensor configurations performed worse than the control. As shown in Table IV, in this scenario the best performing sensor configurations were those that included N, NE, and NW sensors.

Thus reducing the energy consumed from 488 Watts down to the range of 345-360 Watts. In this scenario having only N, NE, & NW sensors performed best, and all sensor-configurations with additional sensors had a higher energy consumption. In this scenario, the number of turns taken by the car varied between 58-62 turns for the best performing sensor configurations, and the number of sensors used varied between 390-1024.

TABLE III. MAP 2 SENSOR CONFIGURATION ENERGY USAGE

Sensor Configuration	Turns	Sensor usages	Dis-tance	Potential Efficiency Ratio	Esti-mated cost (W)
N _	55	84	29	NA	219.44
N _ S _	58	176	30	-0.033	232.56
N _ SE _ SW _	52	252	32	0.018	224.24
N _ S _ E _ W _	50	316	29	0.022	213.35
N _ E _ W _	50	237	29	0.033	210.98
N _ NE _ E _ SE _ S _ SW _ W _ NW _	4	264	29	0.283	91.73
N _ NE _ NW _ SE _ SW _	6	175	29	0.538	94.28
N _ NE _ E _ NW _ W _	6	175	29	0.538	94.28
N _ NE _ S _ NW _	9	152	29	0.676	101.42
N _ NE _ NW _	9	114	29	1.533	100.28

TABLE IV. MAP 1 SENSOR CONFIGURATION ENERGY USAGE

Sensor Configuration	Turns	Sensor usages	Dis-tance	Potential Efficiency Ratio	Esti-mated cost (W)
N _	116	187	71	NA	488
N _ S _	128	396	70	-0.057	523.06
N _ SE _ SW _	118	567	71	-0.005	504.62
N _ S _ E _ W _	106	700	69	0.019	472.23
N _ E _ W _	106	525	69	0.030	466.98
N _ NE _ E _ SE _ S _ SW _ W _ NW _	58	1024	70	0.069	359.2
N _ NE _ NW _ SE _ SW _	58	640	70	0.128	347.68
N _ NE _ E _ NW _ W _	58	640	70	0.128	347.68
N _ NE _ S _ NW _	62	528	70	0.158	354.76
N _ NE _ NW _	60	390	70	0.276	345.4

VI. DISCUSSION

We did the majority of our research in a simulated environment and started our analysis on the ratio between sensors activated and turns taken. For the ratio analysis, we disregard distance travelled, as in the majority of maps, the distance travelled remained constant across the different sensor configurations. Anomalies of 1 or 2 cells in distance travelled could be due to implementation issues with our simulation, as there is no clear correlation between sensor configuration and reduction in distance travelled. Then we used an autonomous vehicle of 1:10^{th} scale, to gather power usage measurements on how many milliwatts of power is used to turn, activate sensors, and move the car forward. We then feed these measurements into the simulated scenarios to get an estimated value for how much energy a car would have used if it takes the same path as the simulation.

To analyze our data, we look at what the minimum sensor cost (C_S) must be in terms of turning cost (C_T) for the sensor configuration to perform better than the control configuration. In Fig. 3 the control robot takes 116 turns and uses its sensors 187 times. The N, NE, NW sensor configured robot takes only 60 turns and uses its sensors 390 times. For the N, NE, NW robot to use less energy than the N robot, 345 <= 488.

In all but scenario 3 there is always a sensor configuration that performs better than the control, given that C_S is some fraction of C_T. As detailed in Table II, it can be seen that no sensor configuration is better than 1 sensor. This is due to the optimal path being the first path calculated, and therefore no sensors can be added to improve the optimal path, detect obstacles on the path early, or reduce the number of turns to get to the goal.

With regards to the path travelled by the robot, it is interesting to note that all robots follow nearly the same path. The exception is when we ran a robot with a map of the obstacle course, then it followed a more optimal path from the start. So, all robots without prior knowledge of the world follow the same path, regardless of the sensors given to them. This was unexpected, as we had assumed that changing the sensor configuration would affect the distance travelled by the robot. By looking at the paths travelled by the robots, we can analyze this situation. In scenarios where the robot is going down a corridor with a dead end at the end, the robot must travel all the way to the dead end to detect that there is no edge it can traverse diagonally. Similarly, while having more sensors allows the robot to detect obstacles earlier, the obstacles on its optimal path need to be detected by one of its front-facing sensors.

VII. Conclusions and Future Work

Our results do not conclusively show that one specific sensor configuration fares best in all scenarios, but they show a general pattern that sensors in the direction of travel grant greater reductions in energy usage. Furthermore, our results show that adding sensors opposite the direction of travel tends to only increase energy usage. We also found that there are scenarios, like Scenario 3, where adding sensors only increases the energy used.

It was surprising to see that the sensor configuration on a car does not significantly alter the path travelled, the difference that did occur was the number of turns each sensor configuration caused the car to take. Due to the similar energy cost of moving and turning, sensor configurations that can reduce the number of turns are able significantly reduce the energy cost. The energy cost of the sensors used in this work is marginal, being two orders of magnitude less than the cost of turning or moving to another cell. Our analysis on the ratio between the energy cost of a sensor vs. the energy cost of turning shows that there would still be significant energy reductions with sensors that use far more energy. The sensor configuration of N, NE, NW had the highest tolerance for sensors that use more energy across all scenarios tested in this project.

Therefore, our results suggest that a higher energy costing sensor would favor the N, NE, NW sensor configuration, even if it doesn't reduce the amount of turns the car takes. Furthermore, the results suggest that a higher energy using sensor that could reduce the number of turns taken by the car, could further reduce the energy usage of the car.

With regards to the future work that could be done based on this initial research, there are areas in the methodology that could be improved. A direct follow up would be to run the entire project without any simulation. Furthermore, this project could be run with a full-size car. This project used sensors placed at 45-degree offsets from each other, with a maximum of 8 sensors for the car, whereas in a future experiment using a larger scale car one could use a smaller increment of offsets between sensors.

We used the sensors with a sensing range of 2 meter which is compatible with the car's frame. In the future experiments and alternative sensors must be considered with better sensing range when applying to larger scale cars. There are other parameters that future work could look at, to see how they affect the energy usage of the car. In particular, this includes the speed of traversal through the course in order to see if the number of sensors and speed of the car affect the energy usage. Future work could also take a look at different path finding algorithms and how they affect energy usage depending on how long the car takes to recalculate its path. Different types of autonomous vehicles may also be investigated. For example, it would be interesting to see how sensor placement would affect a bipedal autonomous robot, an autonomous bike, or a holonomic car.

References

[1] L. Alonso, et al. "Ultrasonic Sensors in Urban Traffic Driving-Aid Systems", Sensors 2011, 11, 661-673; doi:10.3390/s110100661

[2] R. W. Wall, J. Bennett and G. Eis, "Creating a low-cost autonomous vehicle," IEEE 2002 28th Annual Conference of the Industrial Electronics Society. IECON 02, Sevilla, 2002, pp. 3112-3116 vol.4, doi: 10.1109/IECON.2002.1182894.

[3] C. Yan, W. Xu, J. Liu, "Can You Trust Autonomous Vehicles: Contactless Attacks against Sensors of Self-driving Vehicles", ACM ISBN 978-1-4503-2138-9. DOI: 10.1145/1235

[4] J. Borenstein and Y. Koren, "Obstacle avoidance with ultrasonic sensors," in IEEE Journal on Robotics and Automation, vol. 4, no. 2, pp. 213-218, April 1988, doi: 10.1109/56.2085.

[5] W. Xu, C. Yan, W. Jia, X. Ji and J. Liu, "Analyzing and Enhancing the Security of Ultrasonic Sensors for Autonomous Vehicles," in IEEE Internet of Things Journal, vol. 5, no. 6, pp. 5015-5029, Dec. 2018, doi: 10.1109/JIOT.2018.2867917.

[6] Bosch Engineering Center Cluj and BFMC Organizers, "Connection diagrams and components", Repository, https://bfmcmain.readthedocs.io/en/stable/connectiondiagram.html, Viewed on November 30, 2020.

[7] L. D'Alfonso, et al. A polynomial based SLAM algorithm for Mobile Robots using ultrasonic sensors - Experimental Results. 2013 10.1109/ICAR.2013.6766462.

[8] D. Kortenkamp, R. Bonasso, R. Murphy. Artificial Intelligence & Mobile Robots, by, ISBN 0-262-61137-6.

[9] J. Ramsdaleet al. Grid-based mapping: A method for rapidly determining the spatial distributions of small features over very large areas. Planetary and Space Science. 140. 10.1016/j.pss.2017.04.002.

[10] S. M. LaValle, Planning algorithms. New York (NY): Cambridge University Press, 2014.

[11] A, Lluís, "Optimization (Modelling in Science and Engineering)", Link: https://mat.uab.cat/~alseda/MasterOpt/AStar-Algorithm.pdf

[12] S. Thrun, W. Burgard, D. Fox. "Probabilistic Robotics (Intelligent Robotics and Autonomous Agents", in The MIT Press, (2005), ISBN. 0262201623.

[13] E. Weisstein,. "Moore Neighborhood". MathWorld.

[14] E. W. Dijkstra. "A note on two problems in connexion with graphs" (1959). Numerische Mathematik. 1: 269–271.