Course name	**Module name**	**Coordinator**	**Date**
Intelligent Systems and Robotics, MSc	Mobile Robots	Dr Chigozirim Uzor	08/01/2021
Title	Coursework		
Author	Lukasz Pszonak (P2619730)		
Type	University report		

ABSTRACT

The report concerns 5 tasks specified in the Mobile Robots Coursework handout which are finding the middle of the room the robot starts in, leaving the room to find the beacon, stopping by the beacon, returning to the middle of the room and mapping the environment. There are 6 separate sections in the report. The four main ones are: The Introduction that describes the tasks and the software that's being used to fulfill them, Discussion that explains chosen methodology to complete given objectives, System evaluation that will focus on an objective analysis of the simulation then Summary in which all work put into creating this system and report will be recapitulated. The last two sections of this paper are References and Appendixes. According to the module leader, the handout itself leaves a lot of room for students' own interpretations, thus the author came up with some unique solutions for specific tasks. They are explained further in the Discussion section of this paper. The code and the scene are submitted as a zip file. The Appendixes contain screenshots that present various experiments conducted after finishing the main objectives of the coursework.

TABLE OF CONTENTS

INTRODUCTION

The Mobile Robots Coursework requires to utilize developed programming skills with knowledge gained throughout many weeks of lectures and practical sessions over the course of the module. Once again, the author decides to use CoppeliaSimEdu (V-rep) as the simulation environment and lua as the programming language. Given the author's previous experience with MATLAB during his undergraduate studies (Automatic Control and Robotics), he decided, in order to extend his skills and knowledge, that it is best to finish this module using the software and language that were first introduced to him in this semester.

CoppeliaSimEdu is a robot simulator that has integrated development environment. It is commonly used for quick algorithm implementation, factory automation simulations, robotics education (which proved very useful in this module) and much more.[1] The author had no previous experience with this software (before the module teaching commenced), but it proves to be a good choice from the module leader's side because it is relatively easy to learn, the environment is adaptable (this is very important in robotics because it helps the inventor create easier conditions for his testing simulations) and it is not too impactful on the processor's memory. It is worth mentioning that the CoppeliaSim community is vast and there is a forum where many beginners should start their journey with the software.[2]

According to the Coursework handout and specification delivered by the module leader, there are 5 tasks to be fulfilled by the robot during the simulation. These are as follows:

I – The robot starts anywhere in the room, proceeds to find the middle of it,

II – the robot aligns itself with the exit and leaves the room without bumping into walls,

III – the robot proceeds to seeking the beacon (but wanders randomly), which is an object located somewhere in the scene, outside of the room; while looking for the beacon the robot must avoid

[1] https://www.coppeliarobotics.com/features, date and time of access: 05.01.2021 | 17:16.
[2] https://forum.coppeliarobotics.com/, date and time of access: 05.01.2021 | 17.28.

obstacles that are placed at random in the scene, when the beacon is found and recognized, the robot must stop by it (closer than 0.5 m from it),

IV – once the robot has stopped, it must return to the room and stop in the middle of it,

V – while the simulation is running, a map of the environment (scene) should be created.

Picture 1 presents the scene used to complete the coursework. It is a modification of the one the module leader shared via Blackboard. It is important to mention that this is not the final state of the scene, because both the pioneer starting position and the beacon placement differ while various experiments are conducted, as well as new objects and obstacles are added to make the robot's job more difficult.

Picture 1. The scene and its modification for the testing of the robot.

DISCUSSION

After thoroughly reading the specification of the coursework and the handout the author notices that many things are left to his own interpretation[3]. Thanks to it, the work is more demanding and at the same time allows for some out of the box thinking, in which the author feels confident. Hence, in order to make the simulations as efficient as possible (because the testing and experiments are vastly time-consuming), the author decides to exploit the fact that students can interpret some unspecified things in their own way and adds his own ideas and conditions that are going to benefit the simulation. Continuing, to speed up the testing process, the author places the robot in the middle of the room, immediately scans the surroundings and analyzes whether it is the center of the room (if it is, a message: "I am approximately in the middle of the room." is displayed, otherwise it keeps looking) then proceeds to leave the room. It is impossible to be very accurate in determining what the middle is, so the robot approximates, by referring to the actual middle coordinates and checks if it is in very near distance from it. Basing on many conversations during online sessions with the module leader during which students were asking questions regarding the coursework, the author concludes that the robot must wander randomly after leaving the room and finding the beacon should be purely accidental. Which means, the pioneer cannot know where to go, but when it is close to the beacon, it must recognize it. The first concept for this is to use a vision sensor that the author attaches to the Pioneer_p3dx robot but after experiencing difficulties with reading and translating data from it, he brings up another idea, this time with success. Fulfilling the task is possible using sim.CheckDistance() function. The beacon is treated as any other obstacle the robot must avoid however if the mentioned function returns the correct value ("1"), it means that the distance between the pioneer and the beacon is less than 0.51 meters and the robot stops for approximately 5 seconds (the message: "I found the beacon!" is displayed), before returning to random wandering again. The author assumes that the robot must not know where the room

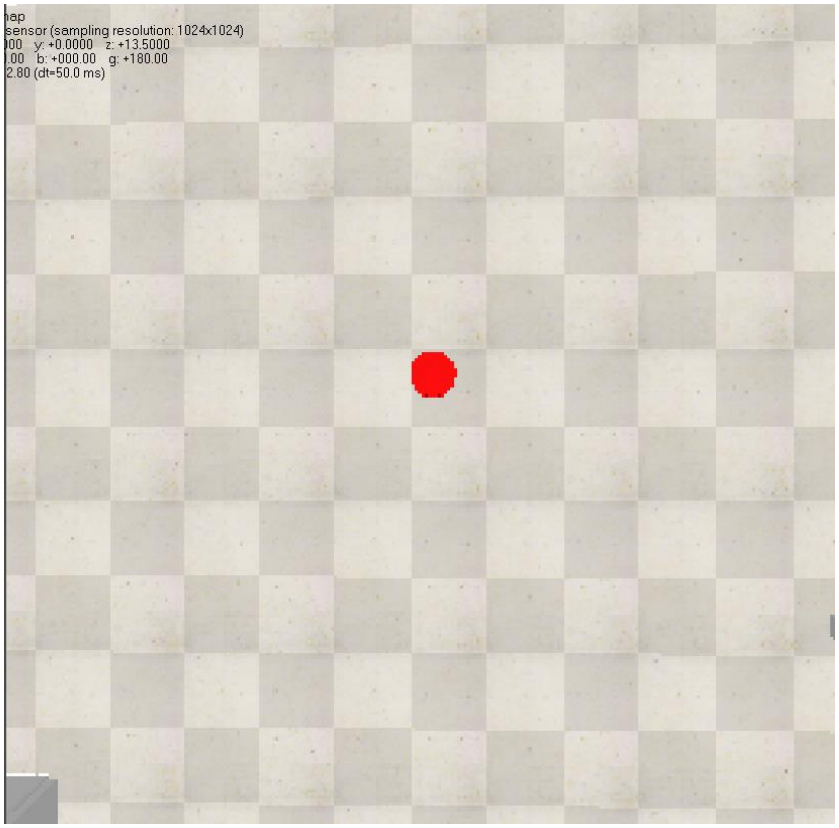
[3] https://vle.dmu.ac.uk/webapps/blackboard/content/listContent.jsp?course_id=_559719_1&content_id=_4893625_1, date and time of access: 05.01.2021 | 18:35.

was and entering it should be also accidental, like with the beacon. This proves that long simulation times are inevitable. For a detailed and objective system evaluation, lots of tests must be conducted and that requires an immense amount of time. To make the robot wander randomly but still avoid obstacles the author decides to mix the Breitenberg function with math.random(). The Breitenberg algorithm is responsible for the avoidance while the other one ensures that when no obstacles are detected the robot appears to be wandering with no specific direction – hence randomly.

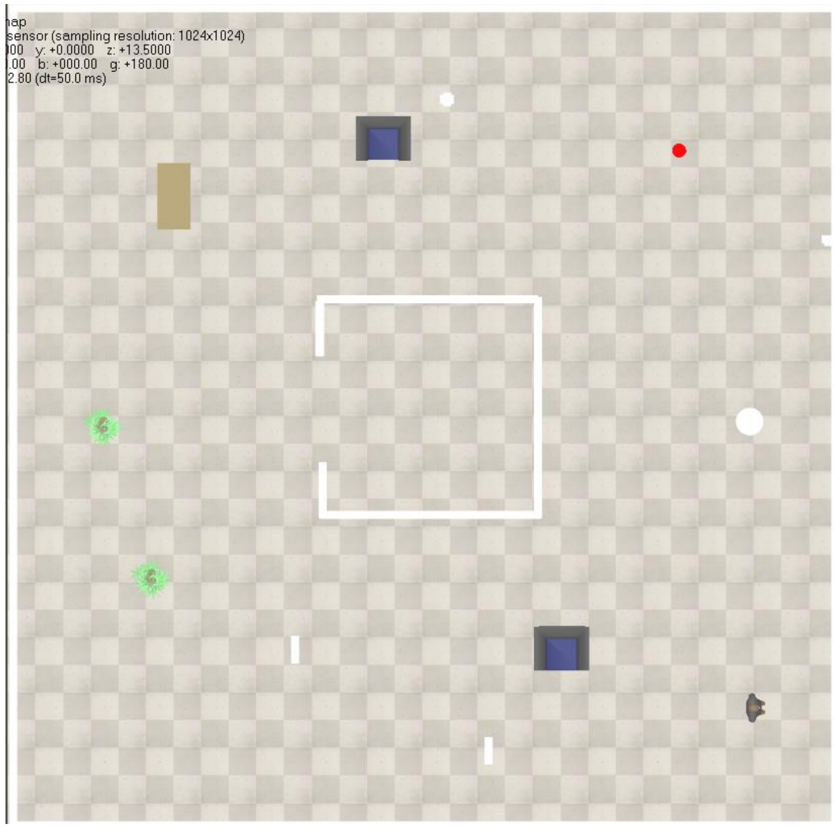
For real-time view and to observe the robot's progress from different perspectives the author adds 3 vision sensors. The first one is attached to the robot and moves with it in the scene displaying what is ahead of it. The second sensor is located above the beacon and it is visible when the robot is close and stops/moves by it, and the last one is located above the map and is given a higher resolution in order to map the environment entirely and accurately. The author is confident this is the best and "out of the box" approach to the mapping task because it is error-free in comparison to other mapping methods.[4] Pictures 2, 3 and 4 respectively show views from the mentioned vision sensors.

[4] Gregory F. Ivey, Eric N. Johnson, *Investigation of Methods for Simultaneous Localization and Mapping Using Vision Sensors*, 2006.

Picture 2. View from the vision sensor on the pioneer.

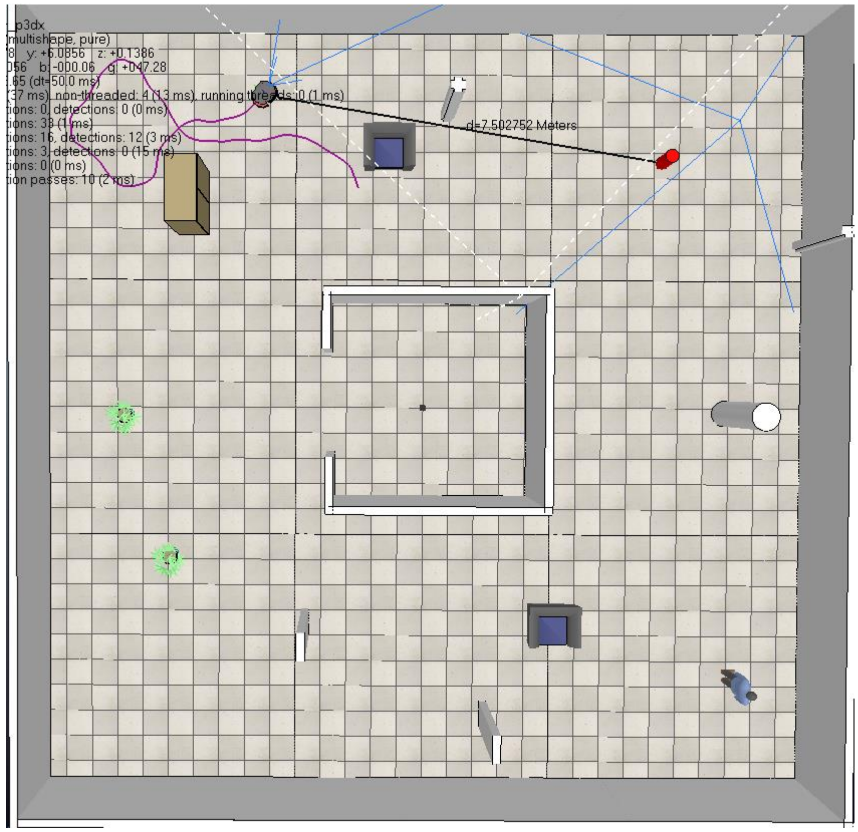


Picture 3. View from the vision sensor above the beacon (red object).



Picture 4. View from the vision sensor above the scene. Map of the environment.

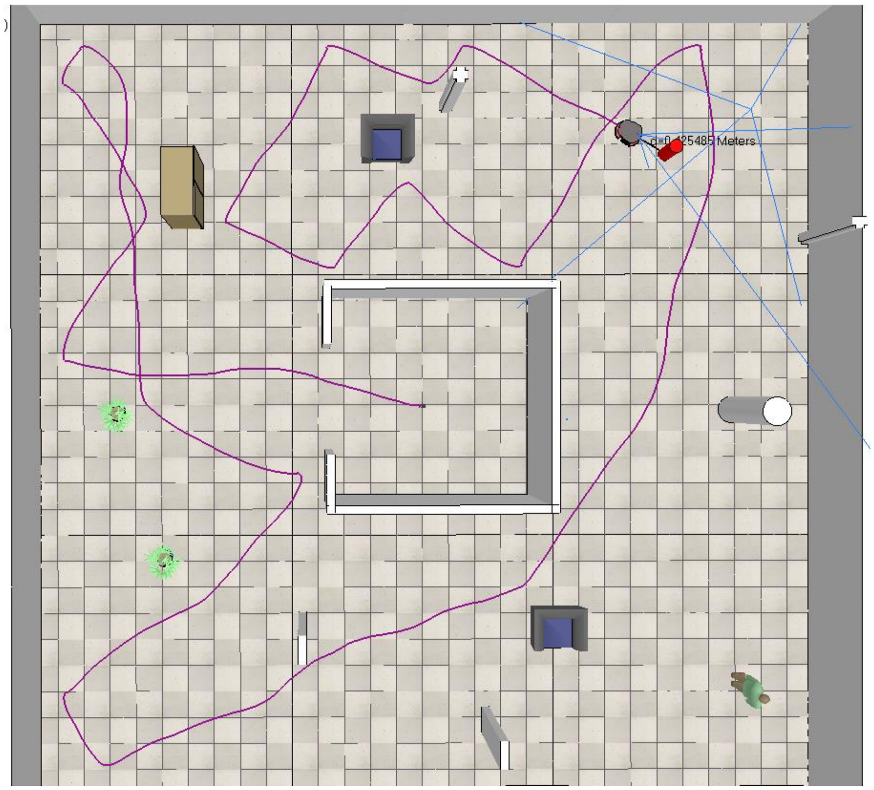
Picture 5 is used to show the robot's behavior while looking for the beacon. It avoids obstacles and wanders randomly. It is visible that the robot is leaving a path after itself. Later in the testing the author decides to extend the path memory so it is visible from the beginning to the end of the simulation.



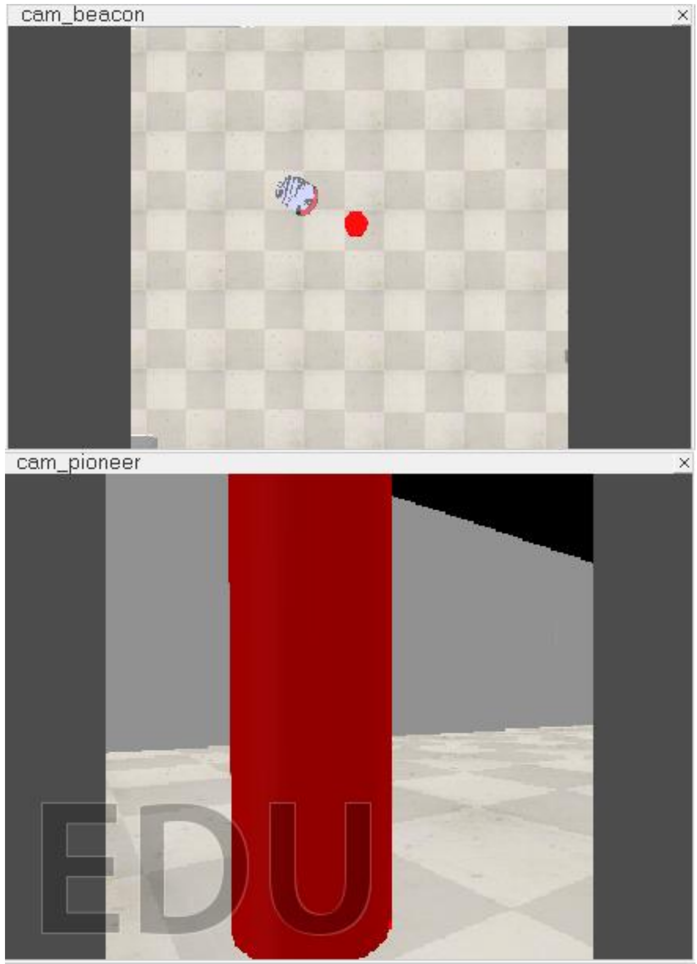
Picture 5. Obstacle avoidance + random wandering.

Pictures 6 presents a simulation state when the robot stops by the beacon, and Picture 7 displays views from 2 vision sensors - the one attached to the pioneer and the one above the beacon. Using these vision sensors adds more realism to the simulation[5], which is helpful since the author is currently unable to conduct the testing on the real robot.

[5] David Weikersdorfer, Raoul Hoffmann, Jörg Conradt, *Simultaneous Localization and Mapping for Event-Based Vision Systems*, 2013.



Picture 6. Robot stops by the beacon.

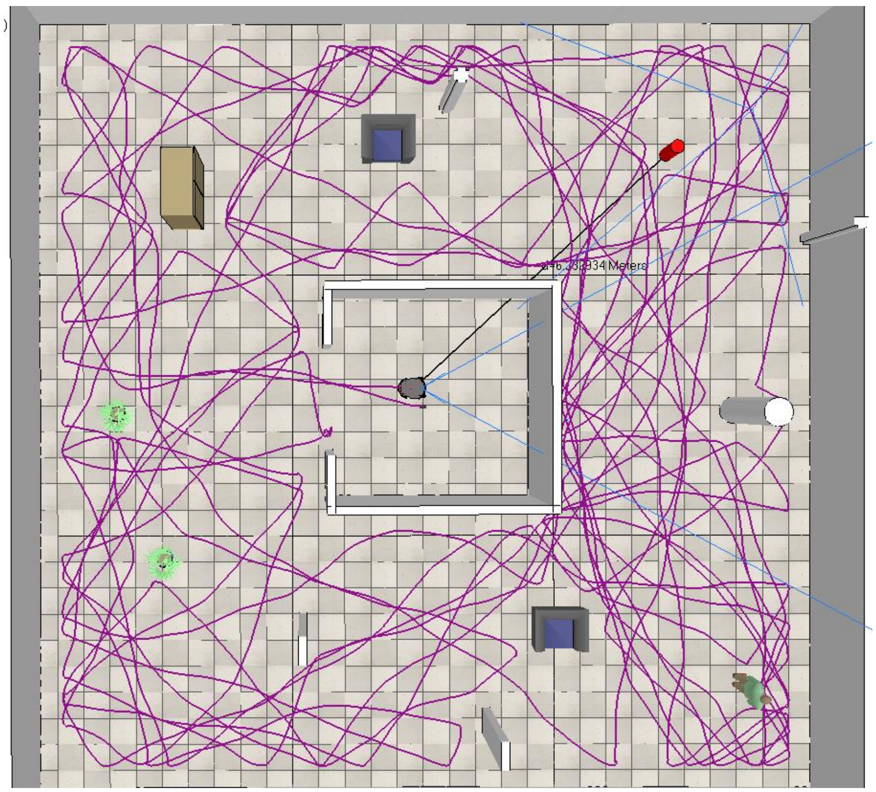


Picture 7. View from the pioneer and beacon's vision sensors while the robot is not moving.

Because of the random wandering aspect in the simulation the author predicts that a situation where the robot approaches the beacon more than once is possible, hence in order to make the robot not stop again by it, he adds a counter both for beacon and middle detection. The robot can recognize the beacon only once, on another occasion it is not recognized and is treated as any other obstacle.[6] He avoids it. Because one of the objectives of the coursework is to find the middle as well as end the simulation there, the counter for the middle has a higher value and can be detected up to 2 times - once when the simulation starts and once when it ends. On Picture 8, the final state of the simulation is displayed. The robot manages to find the room and stops approximately in the middle of it. It is visible that the path it leaves behind itself is very long. It is

[6] David Lowe, *Distinctive Image Features from Scale-Invariant Keypoints*, 2006.

caused by the random wandering aspect of the simulation and the fact that the robot must find the room, which is interpreted as "the robot does not know what or where the room is". Finding it is accidental which is in accordance with the coursework handout. In order to shorten the time needed to get the robot to the room again one could use path-planning[7], but it is against the specification of the coursework.



Picture 8. End of simulation. Robot is back in the room and approximately in the middle of it.

[7] https://medium.com/@afakharany93/path-tracking-tutorial-for-pioneer-robot-in-vrep-acffcf76875b, date of access: 06.01.2021 | 16.29.

SYSTEM EVALUATION

To evaluate the system the author decides to use various configurations of robot and beacon positions, add/remove some obstacles and even remove all the obstacles in one configuration (only beacon remains). Adding and removing objects from the scene is done in order to find whether there is a relation between the number of obstacles and the simulation time. Pictures presenting various configurations are added in the Appendixes section of the report. Table 1 shows the comparison between average simulation times for various configurations of the system. The factor took into consideration here is the number of obstacles, not their positions, however they may affect those times indirectly. When new obstacles are added, positions of the other ones remain the same. Simulation times are taken as averages from 5 successful simulations. The default number of obstacles are 12 (11 + the beacon). This number of obstacles was used in the testing of the system, to see if all tasks can be fulfilled successfully.

Table 1. Comparison between simulation times for different number of obstacles.

Number of obstacles (beacon is included)	Simulation time 1	Simulation time 2	Simulation time 3	Simulation time 4	Simulation time 5	Average time
12 (default)	15 minutes 23 seconds	17 minutes 4 seconds	11 minutes 59 seconds	24 minutes 27 seconds	19 minutes 33 seconds	**17 min 41 sec**
15	7 minutes 8 seconds	7 minutes 56 seconds	9 minutes 44 seconds	16 minutes 31 seconds	33 minutes 9 seconds	**14 min 53 sec**
25	8 minutes 19 seconds	11 minutes 12 seconds	9 minutes	13 minutes 13 seconds	13 minutes 44 seconds	**11 min 5 sec**
8	14 minutes 46 seconds	15 minutes 57 seconds	15 minutes 40 seconds	11 minutes 8 seconds	12 minutes 11 seconds	**13 min 56 sec**
4	11 minutes 56 seconds	21 minutes 55 seconds	7 minutes 17 seconds	20 minutes 42 seconds	15 minutes 17 seconds	**15 min 25 sec**
1	15 minutes 11 seconds	28 minutes 33 seconds	41 minutes 6 seconds	30 minutes 38 seconds	13 minutes 4 seconds	**25 min 42 sec**

After a detailed analysis of Table 1, the author notices that there is a relation between the number of obstacles and the simulation time. According to the results, the more obstacles are placed in the scene, the shorter is the average time of the simulation. He comes up with a conclusion that the random wandering aspect can also affect this but makes it impossible to determine whether the relation between the simulation times and the number of obstacles is direct or indirect.[8] In order to be sure, random wandering must be removed, but it stands against the specification of the coursework thus the author decides to omit it.

Comments:

1) The simulation is not always successful; the author is unable to find a suitable replacement for the sim.setObjectOrientation() function that is used to align the robot with the exit of the room, and this function is a bit defective. There are times when the robot finds the center of the room, however this function detaches the wheels off the pioneer, and it is unable to continue. The simulation must be restarted then.

2) The most efficient way for the simulation to run successfully is to place the robot in the middle of the room (it will realize it is the middle when the simulation starts).

3) If there is no aligning error (the simulation has about 95% chance to be completed successfully, given time. Because there is no time limit for the robot to find the beacon and the return to middle of the room, the simulation (in theory) can last forever or until it fails.

4) Unless the beacon's position is really hidden or surrounded by lots of obstacles, the robot can find it.

5) Going back to the middle of the room takes most of the simulation time.

[8] Adamson, Kardong-Edgren, & Willhaus, *Review of evaluation instruments*, 2013.

SUMMARY

The coursework proves demanding and requiring lots of time. Throughout the testing phase the author tries many different approaches to problems however only those mentioned in the report are successful to some degree. It becomes clear to the author that creating an error-free system for 5 different tasks such as these mentioned in the report is an immense effort for a beginner, however he is satisfied with the result because, most of the time, the scope of the coursework is fulfilled. Some adjustments are needed to correct certain aspects of the simulation but with determination and time, the author is confident he can make the system work almost perfectly. For future improvements of the code (outside the scope of this coursework), the author considers implementation of path-planning that would make the robot return to the room faster. A* planning is identified as the best solution.[9] To sum up, given all the time and resources that are provided for students to complete the coursework, the author is confident this module greatly improves his programming skills and extends his knowledge regarding the subject of mobile robots.

[9] http://correll.cs.colorado.edu/?p=965, date and time of access: 06.01.2021 | 20.23.

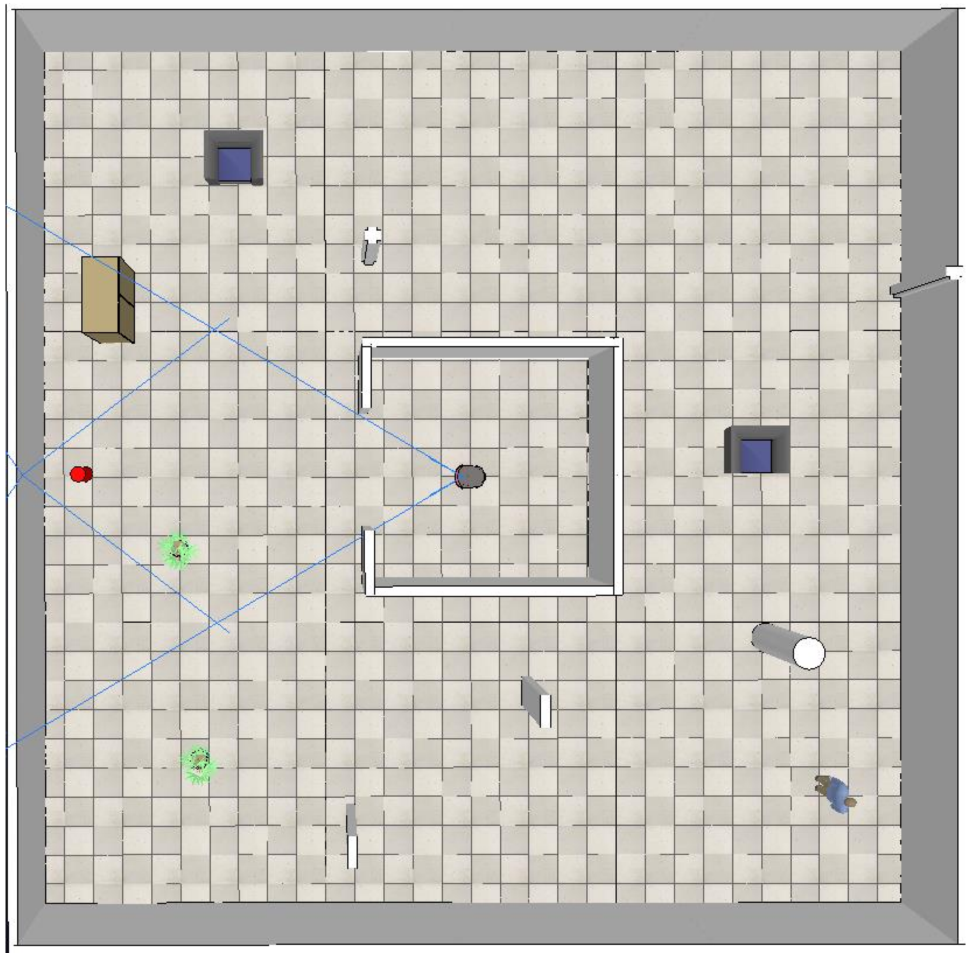
REFERENCES

1. https://www.coppeliarobotics.com/features, date and time of access: 05.01.2021 | 17:16.
2. https://forum.coppeliarobotics.com/, date and time of access: 05.01.2021 | 17.28.
3. https://vle.dmu.ac.uk/webapps/blackboard/content/listContent.jsp?course_id=_559719_1&content_id=_4893625_1, date and time of access: 05.01.2021 | 18:35.
4. Gregory F. Ivey, Eric N. Johnson, Investigation of Methods for Simultaneous Localization and Mapping Using Vision Sensors, 2006.
5. David Weikersdorfer, Raoul Hoffmann, Jörg Conradt, Simultaneous Localization and Mapping for Event-Based Vision Systems, 2013.
6. David Lowe, Distinctive Image Features from Scale-Invariant Keypoints, 2006.
7. https://medium.com/@afakharany93/path-tracking-tutorial-for-pioneer-robot-in-vrep-acffcf76875b, date of access: 06.01.2021 | 16.29.
8. Adamson, Kardong-Edgren, & Willhaus, Review of evaluation instruments, 2013.
9. http://correll.cs.colorado.edu/?p=965, date and time of access: 06.01.2021 | 20.23.

APPENDICES

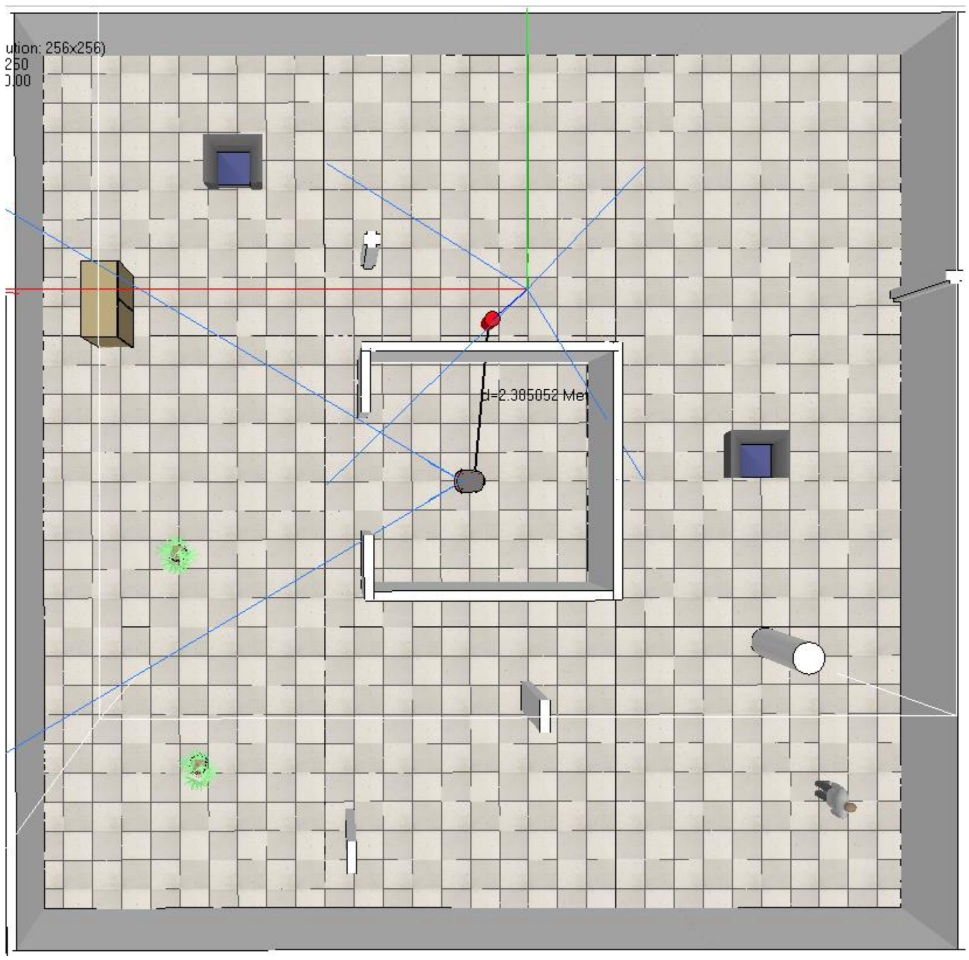
Different robot's/beacon's position configurations:

Picture 9. Configuration 1.

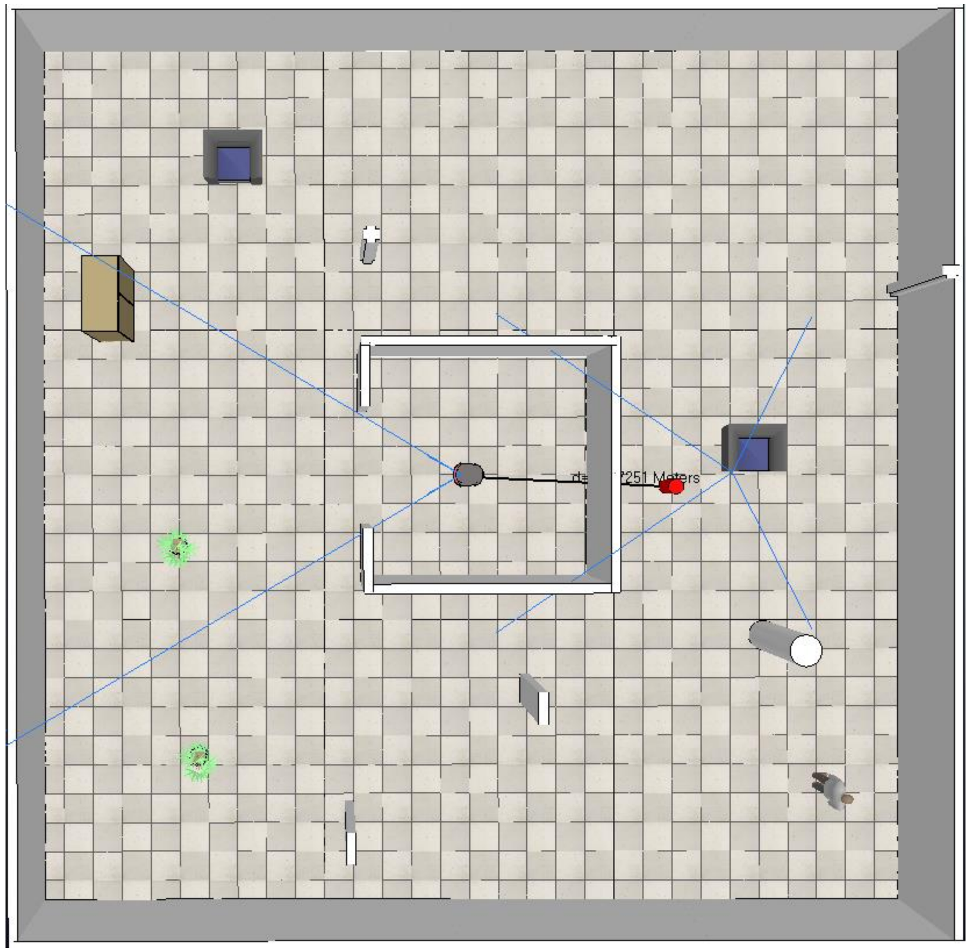


Picture 10. Configuration 2.

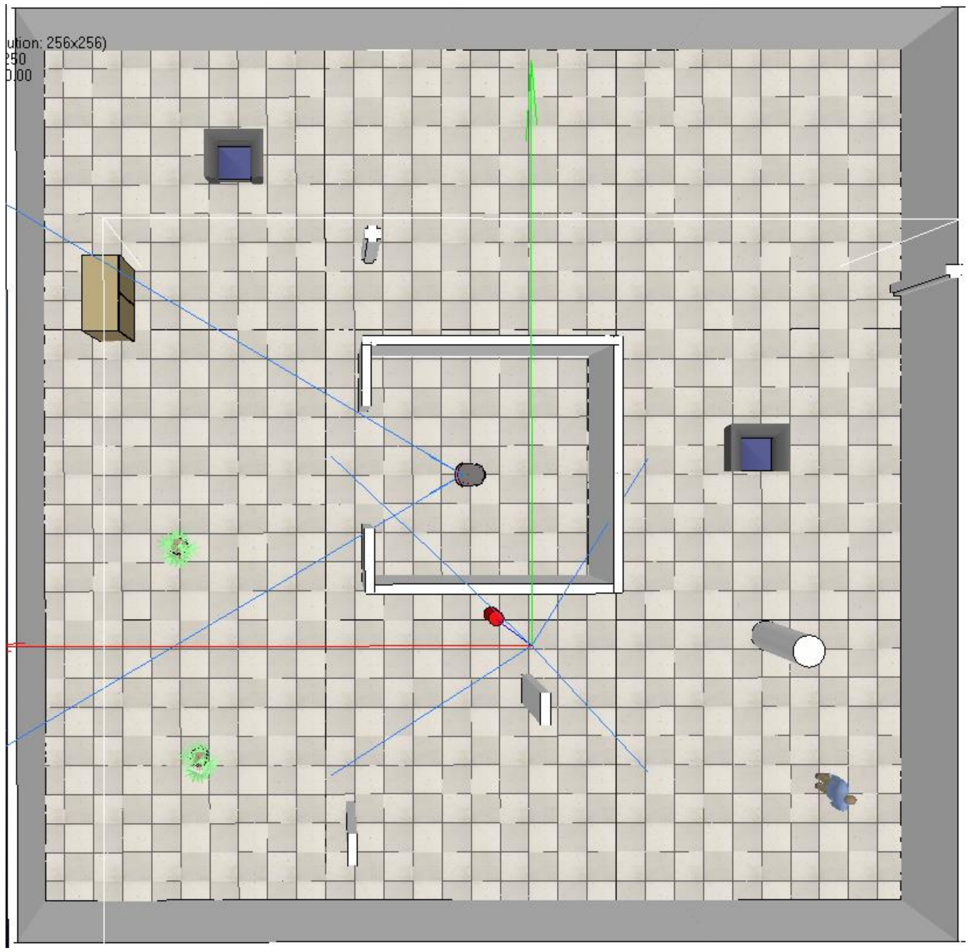
Picture 11. Configuration 3.



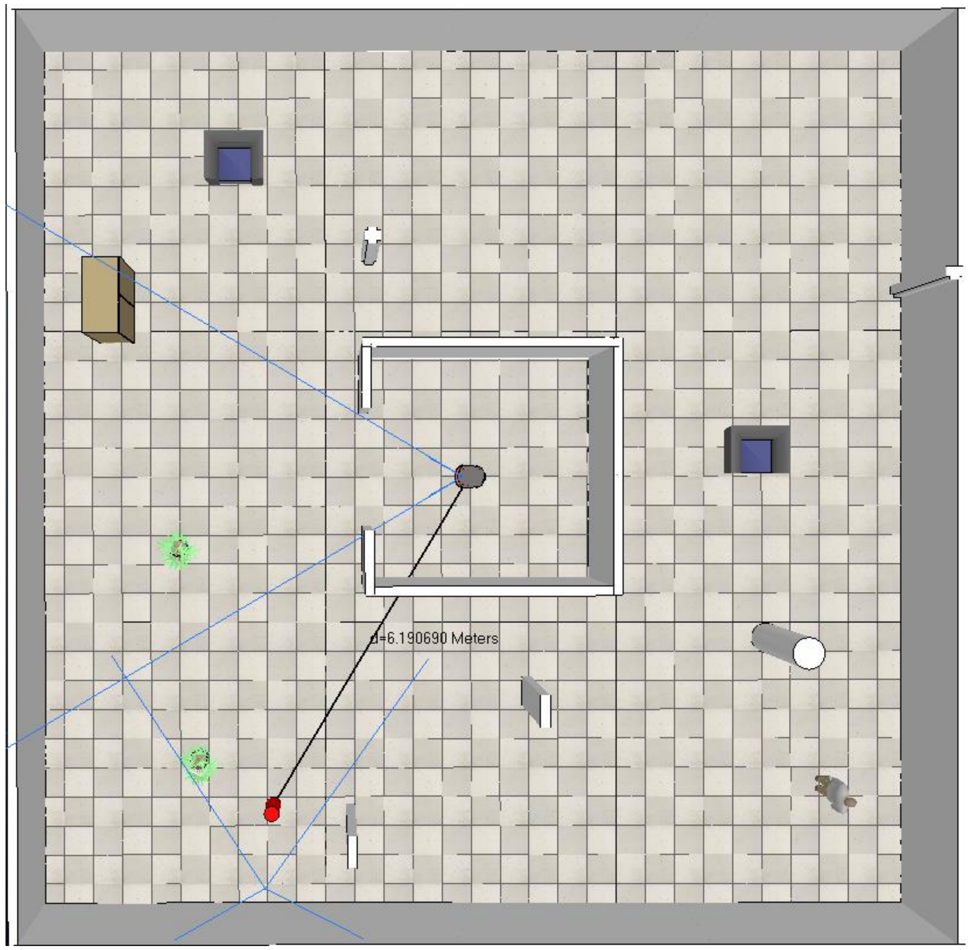
Picture 12. Configuration 4.



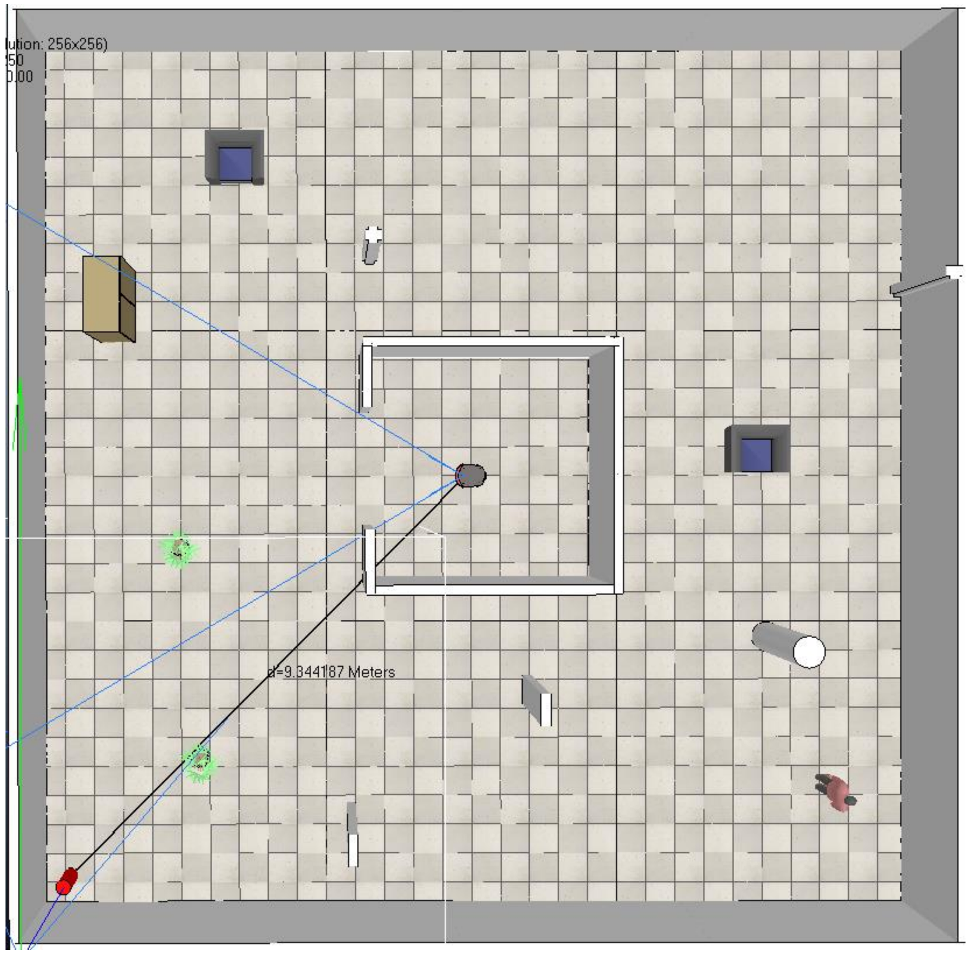
Picture 13. Configuration 5.



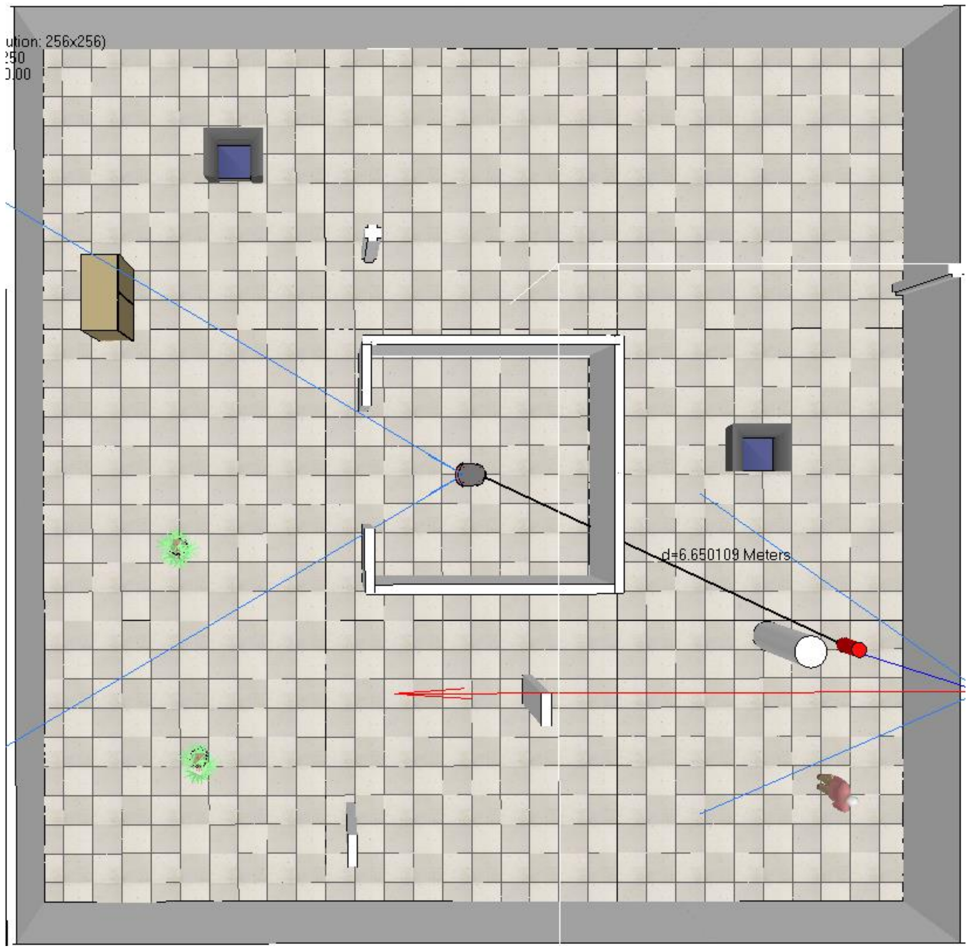
Picture 14. Configuration 6.



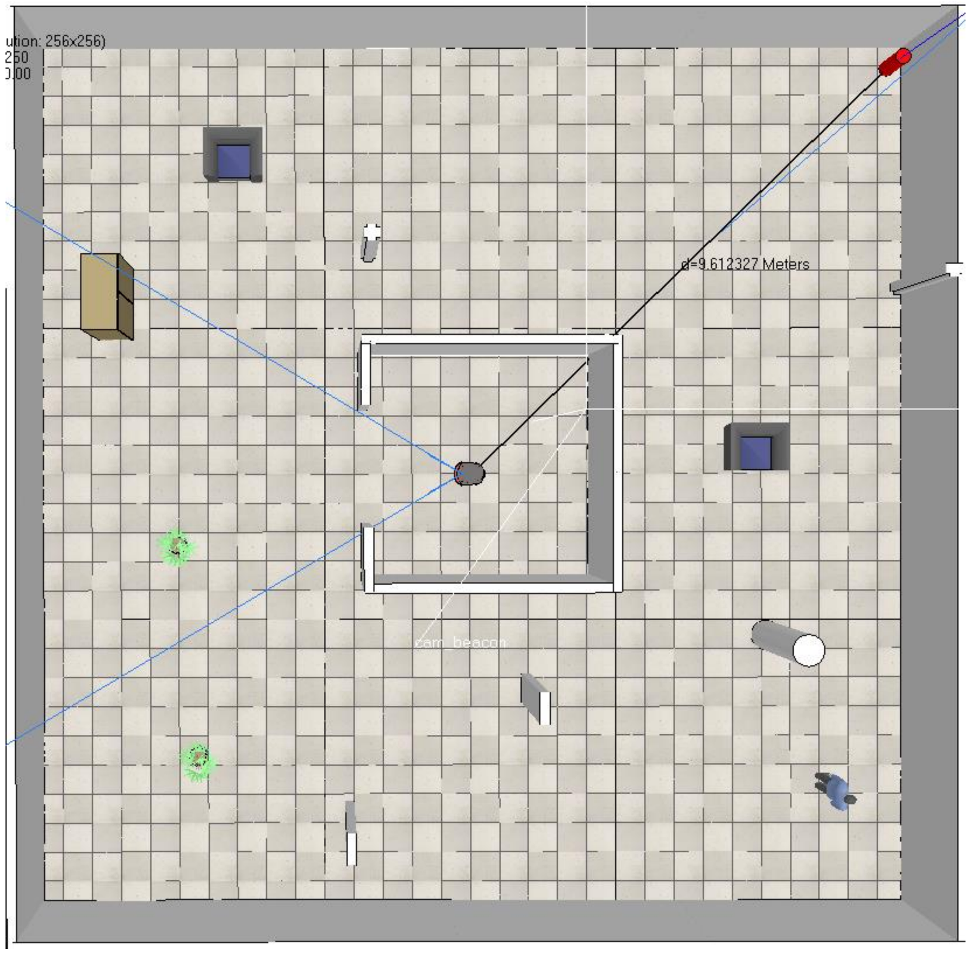
Picture 15. Configuration 7.



Picture 16. Configuration 8.

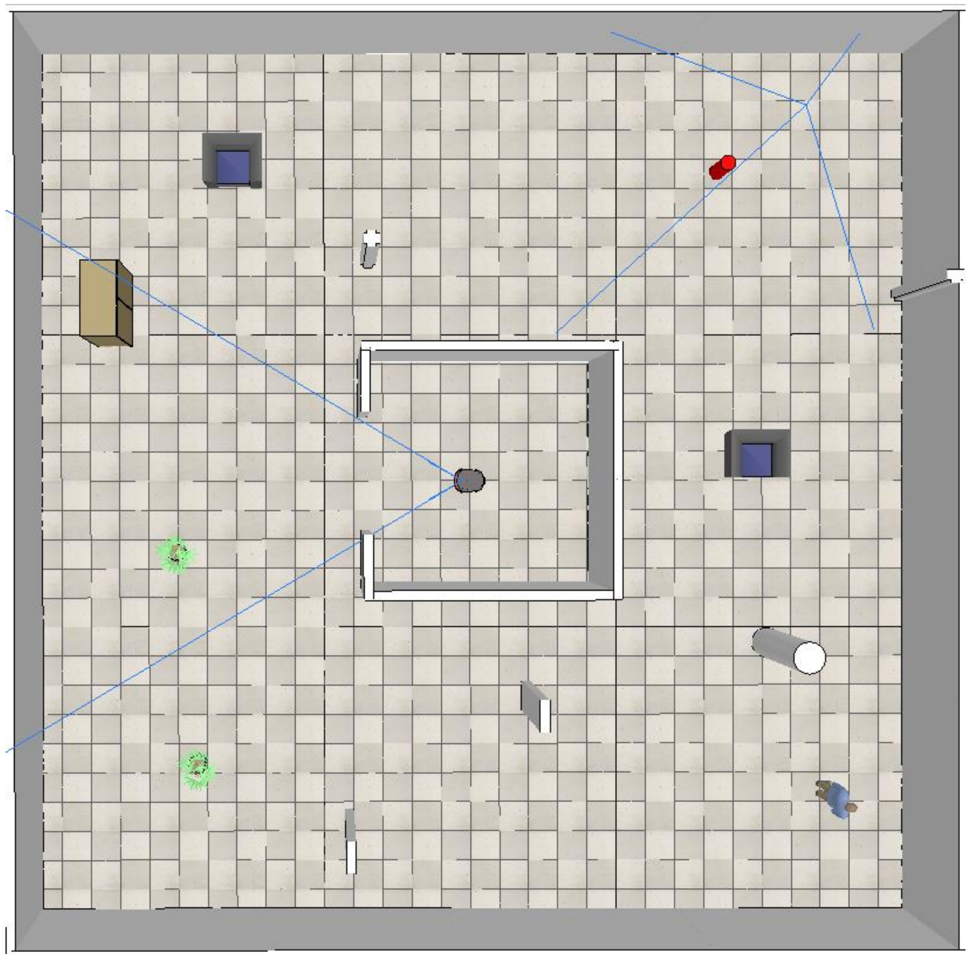


Picture 17. Configuration 9.

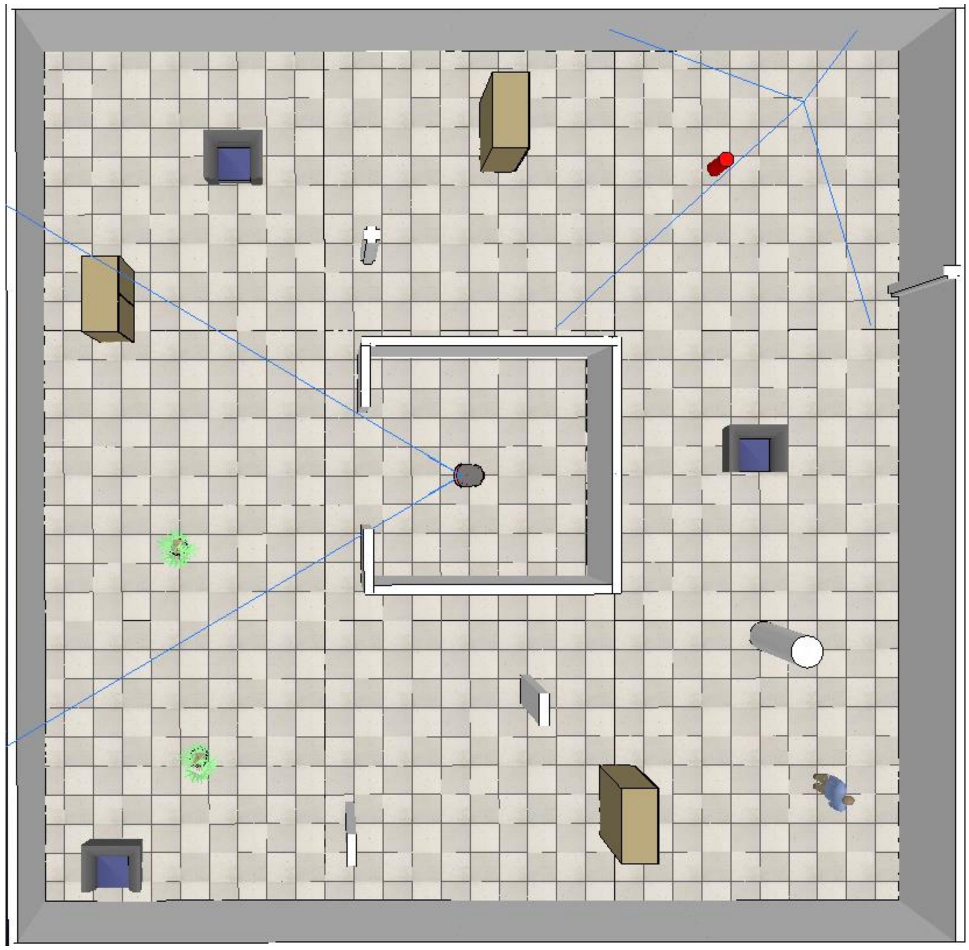


Picture 18. Configuration 10.

Changing the number of obstacles, while the beacon position remains the same:

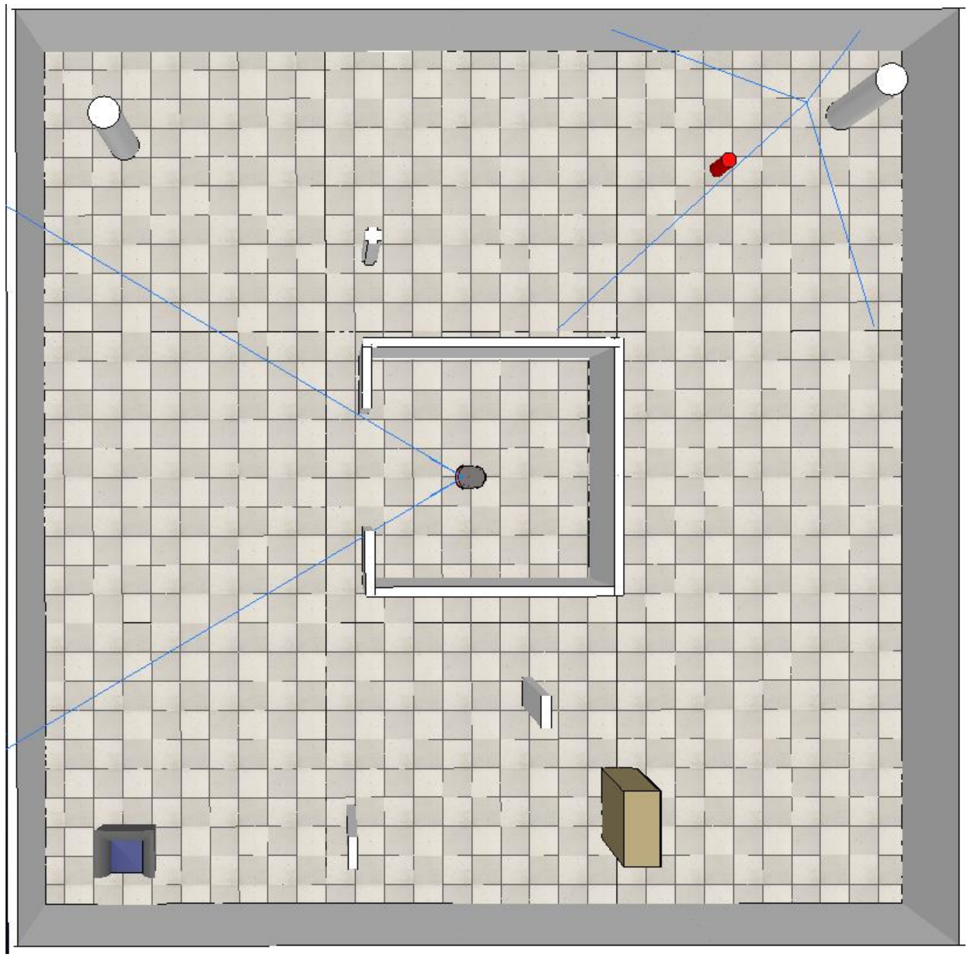


Picture 19. There are 12 obstacles (including the beacon).

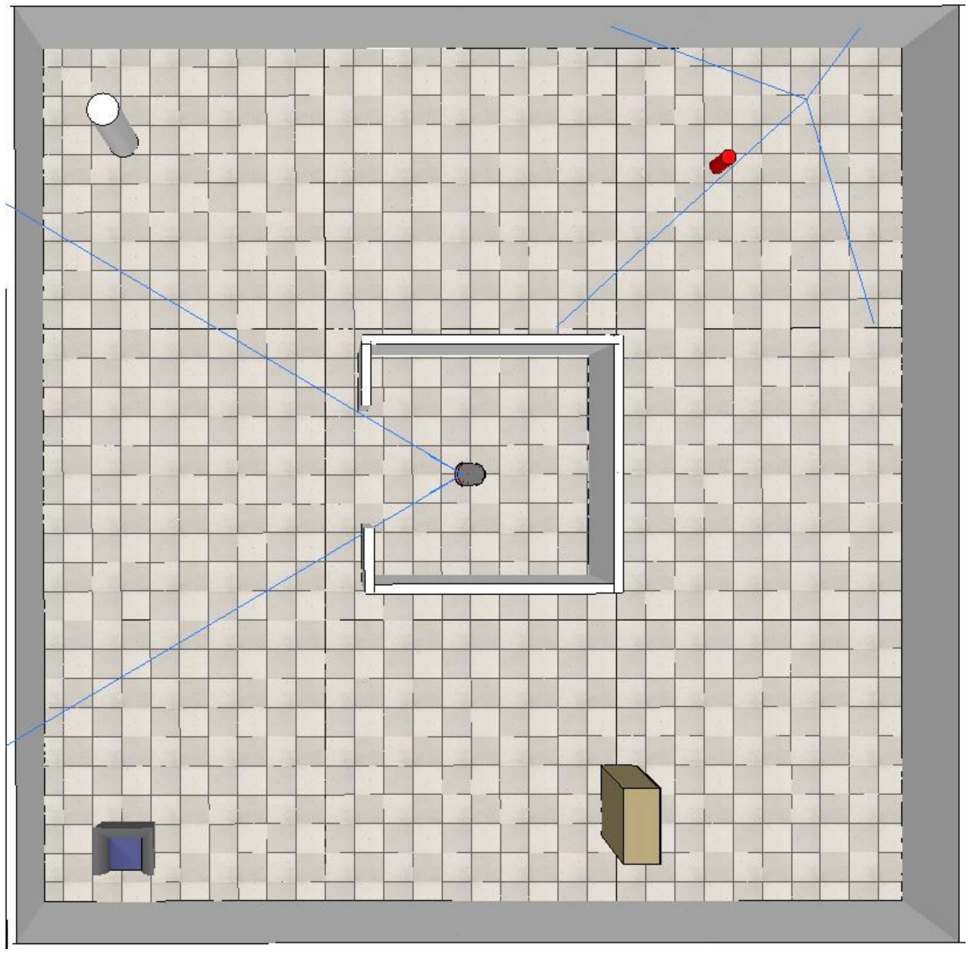


Picture 20. There are 15 obstacles (including the beacon).

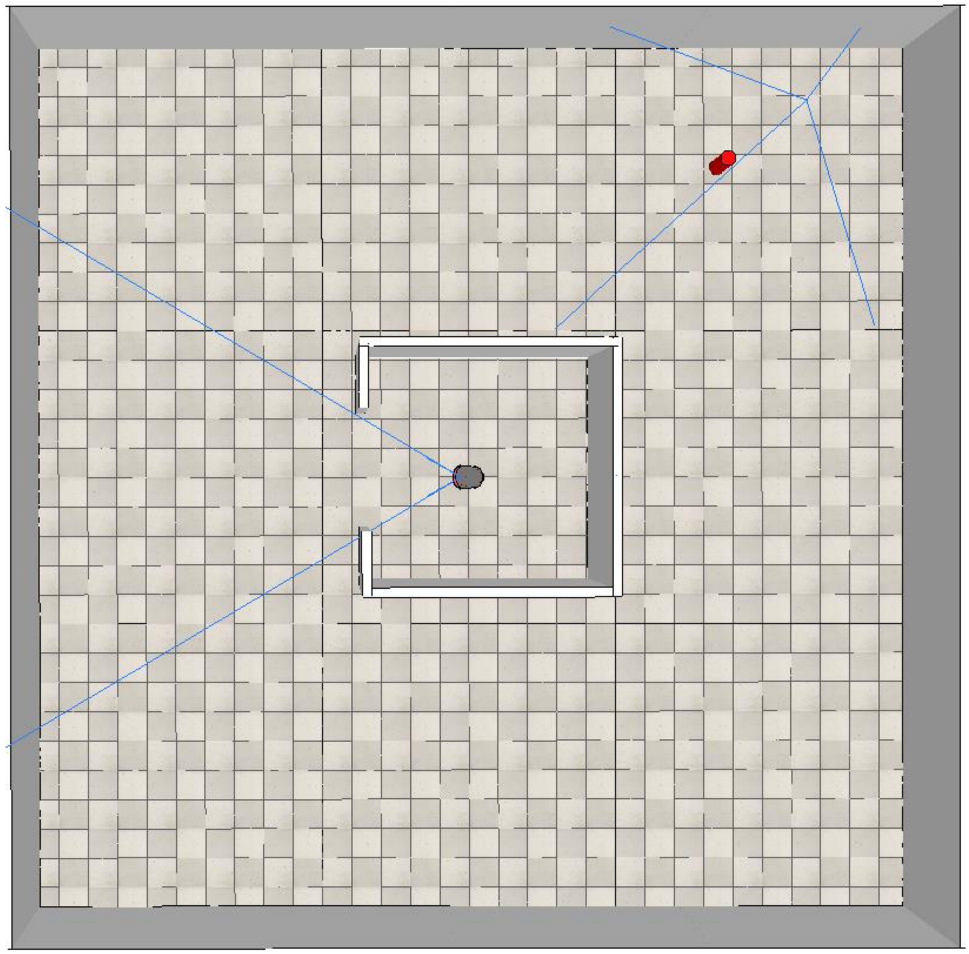
Picture 21. There are 25 obstacles (including the beacon).



Picture 22. There are 8 obstacles (including the beacon).



Picture 23. There are 4 obstacles (including the beacon).



Picture 24. There is 1 obstacle - the beacon.