Tour Guide Bot

BLG456E Project Proposal

Team

Marvin the paranoid android

17.10.2018

Recep Can Babaoğlu -150130112
Eray Mert KAvuk - 150130133

Table of Contents

Introduction

Exhibition guide TurtleBot will design for deal with a real issue. As we all see finding a guide or reaching the necessary information is a problem for many people when they visit an exhibition, art gallery or a museum. In this project we would like to do something can be pioneer in this area.

In this project, we are going to design and develop Exhibition Guide Robot for primary school children who are just start learning about math and geometry. In our exhibition we will exhibit some geometric shapes such as triangle, circle etc…Our aim is make children love math and geometry. That's why we are trying to make our exhibition funnier and use a turtlebot instead of a math teacher.

The project is interesting because it is a solution for a real issue and also solve it in a funny way that children love it. Our other motivation is also it can be used for many different area as well and it can be improved by others. Not only for children or in a funny way but also it can be used for an art gallery. Especially if the robot supported with enough data that contain information about artworks. It can be wonderful with the language and speech support also. This points are also increase our motivation about our projects.

Realization of the prototype for given purpose will be covered in the report. In this document we will discuss the importance and the beautiful parts of this project, the basic operations our robot will perform, a reasonable schedule for our team members, the methods we will use and a satisfying conclusion.

Background

The main aim of the project is make explanation in the exhibition funnier and in more interactive. We are doing this in order to not bored people who are visiting exhibitions. Our robot can recognize artworks in the exhibition. When it recognize an artwork, it will go to the artwork and stop there a bit. This stop is represent the explanations.

We will implement our robot only in a simulation. We will use ROS environment and Gazebo to simulate our project. We will use Sketch Up for design our environment.We are also planning to use Open Cv library for object recognition. For better understanding we are also planning to follow couple ROS tutorials such as usb_cam, cv_bridge.

Robot will start from at the beginning of the exhibition and then start travel in the exhibition hall. It will try to recognize the artworks and when it recognize them it will move to them. Basically robot will use it's own laser to recognize walls etc and also using object recognition it can recognize the artworks.

When robot recognize an artwork. It will go to it and stop there for couple seconds to give information about artwork. It will be more effective with a good exhibition hall design and we will also try to do that in our project. At the end robot will leave the exhibition hall after it covered all the artworks. As we state above in this project our artworks will consist geometric shapes and we are aiming to make this exhibition for primary school children.

Methods

The software for the turtlebot will be designed in an Ubuntu 16.10 environment with the middleware software known as ROS (Robot Operating System) Kinetic. We will use Gazebo as our stimulator and Rviz as our 3D visualizer.

The robots will have mainly 4 tasks:

- Go to paintings on the wall which hanging predetermined locations.
- Exploring the environment in the direction that the user decided while avoiding any obstacles.
- Recognizing the exhibition area.
- Successfully complete the predefined movements considering the paintings type.

Each task will have its own percept and response sequence, and will be evaluated separately. First and second task are related each other. Both of them trying to take the turtlebot exhibition area without any crash. For the first task we will make use of the turtlebot's distance sensor to perform a series of simple movements. For the second task turtlebot should find the exhibition area. The exhibition area predefined from user.

The first and the second tasks are considering first work package of the project.To measure the success of this tasks we will start with 0 points:

- If the turtlebot successfully reach the exhibition area: +50 points
- If the turtlebot crashes an obstacle, each crash is -10 points.
- If the turtlebot crashes a wall each crash is -10 points.
- If the turtlebot does not reach the exhibition area (wrong coordinates): -20 points.

We will experiment under this point system 10 times and if the overall average is greater than 30 points, we will label the experiment successful.

For the third and fourth task are also related, the turtlebot should recognize the exhibition site. The location of exhibition site will be predefined. Also exhibition area is a straight corridor, there are paintings on the walls. The paintings are composed easily understandable objects such as triangle, square etc. When the turtlebot reach the exhibition area, it should look the walls. The turtlebot makes different movement for each different paintings.

The third and the fourth tasks are considering as a part of second work package.To measure the success of this tasks we will start with 0 points:

- If the turtlebot successfully finish the tour : +50 points
- If the turtlebot does not understand a shape of paintings or does wrong movements each wrong -10 points.

We will experiment under this point system 10 times and if the overall average is greater than 30 points, we will label the experiment successful.

Once the individual tasks reach their acceptable performance which are at least 30 points we will integrate the module and run 10 trials in random exhibition environment with random parameters. If the average of trials be successful, we will think ready for demo.

Schedule & Outcomes

A summary of tasks and their delivery times can be found in Table 1. Below that the tasks are detailed.

Table 1: Main tasks and responsibilities

Task	Responsible Person	Start date	Finish date	Amount of time expected (hours)	Share of marks (%)
WP1: Work Package 1		25 Oct	26 Nov	32	40%
WP1.1: Environment	Can	25 Oct	10 Nov	8	10%
WP1.2: Sensors	Eray	25 Oct	10 Nov	8	10%
WP1.3: Movement	Eray	11Nov	26 Nov	8	10%
WP1.4: Mapping	Can	11 Nov	26 Nov	8	10%
WP2: Work Package 2		27 Nov	24 Dec	50	60%
WP2.1: Path finding	Can	27 Nov	9 Dec	10	10%
WP2.2: Path following	Eray	27 Nov	9 Dec	10	10%
WP2.3: Area Detection	Can&Eray	10 Dec	24 Dec	15	20%
WP2.4: Image Detection	Eray&Can	10 Dec	24 Dec	15	20%

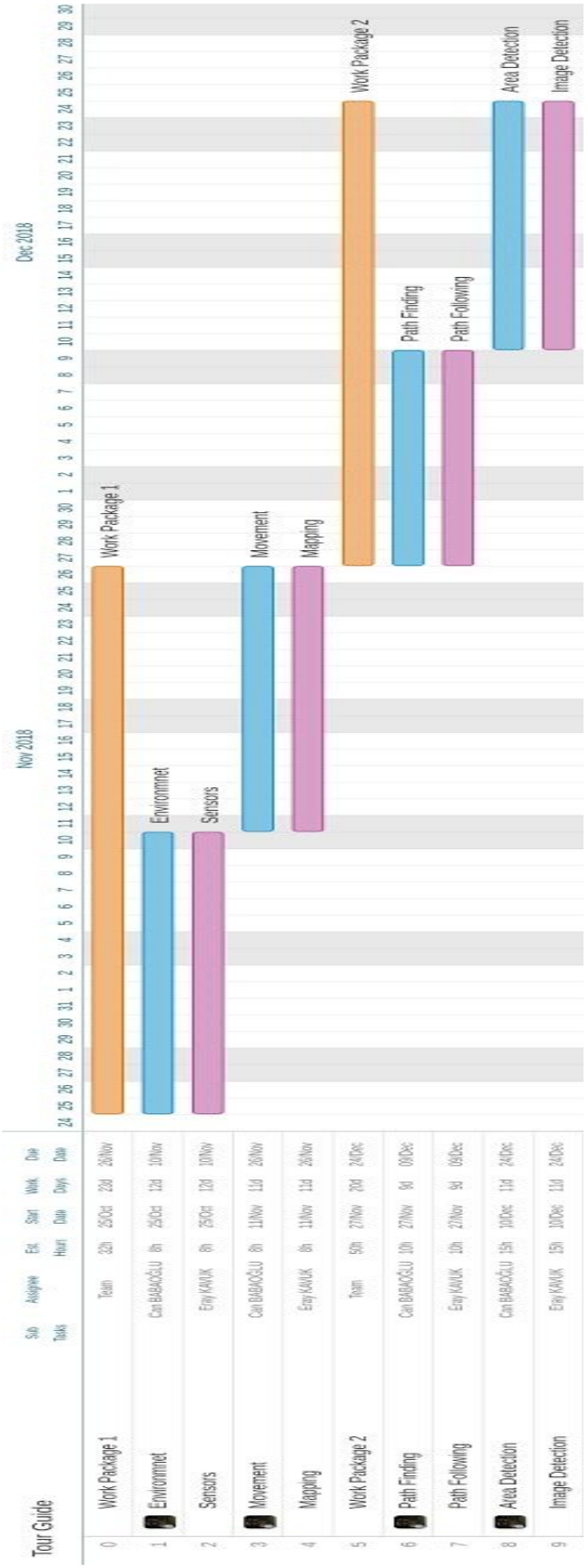

Chart 1: The gantt chart for the tasks. Each color represents a different team member.

Task Definitions:

WP1.1: Environment

- The objective of this task is designing and developing the environment. It includes creating obstacles,exhibition ares and some painting on the wall(at least 3 paintings).
- This task requires nothing to be completed.
- Completion will be determined by checking all the needed components in the environment.
- The main components are paintings. The paintings should be different geometric shapes such as triangle, square, circle.

WP1.2: Sensors

- The objective of this task is understanding the sensors for the robot according to our mission.
- This task requires nothing to be completed.
- Some tests are should be made for understanding data coming from sensors.
- Completion will be determined by the data coming from sensor is sufficient and healthy.

WP1.3: Movement

- The objective of this task is making the robot able to move in 2D space properly.
- This task requires nothing to be completed.
- Some tests are should be made for understanding robot movement.
- Also different movements should be tested for different images.
- Completion will be determined by able to move the robot to all directions and ways.

WP1.4: Mapping

- The objective of this task is mapping the environment using the data gathered from sensors.
- This task requires both sensors (WP1.2) and movement (WP1.3) to be completed.
- Also it should be tested on environment from WP1.1.
- Completion will be determined by analyzing what robot sees around it by rviz.

WP2.1: Path finding

- The objective of this task is finding the shortest path according the map the robot gathered. It includes pathfinding algorithm and stop the movement if an obstacle is discovered.
- This task requires both mapping (WP1.3) and environment (WP1.4) to be completed.
- Completion will be determined by how robust and fast the robot is choosing its path.

WP2.2: Path following

- The objective of this task is going to target using the shortest path according path finding algorithm.
- This task requires both mapping (WP1.3) and environment (WP1.4) to be completed.
- Completion will be determined by how fast the robot reach target without any crash.

WP2.4: Area Detection

- The objective of this task is determining the given area which the robot will starts to image recognition.
- This task requires the WP1 to be achieved.
- Completion will be determined by the success rate of reaching the area correctly.

WP2.5: Image Detection

- The objective of this task is understanding the different painting on the wall.
- This task requires the area detection (WP2.4) to be completed.
- Completion will be determined by investigating accuracy of image detection.

After we have completed all the tasks in project, our robot will go to exhibition area without any crash, detect images and make the different movements correctly.

Conclusion

In our project we are trying to help people in exhibition areas. Traditional methods might take human effort and other solutions such as headphones and etc is can be distractor. Turtlebot will go to exhibition area then will give information about exhibited things and it makes it more interactive and funny. Robot will do it our path finding algorithms and object recognition processes.

Not only a fix a real world problem and also start something new make us more excited about project. When we will see that on the news reel robots guides people in the galleries and the museums, probably we will watch it with a great smile on our faces.

Simulation will be done with ROS~Kinetic[1], Gazebo[2] and Rviz[3] on Ubuntu 16.04.

We are using turtlebot[4] for our simulations.

Reference List

[1] *ROS.org | Powering the world's robots*. (2017). *Ros.org*. Retrieved 22 October 2017, from http://www.ros.org/

[2] Gazebo. (2017). Gazebosim.org. Retrieved 22 October 2017, from http://gazebosim.org/

[3] *rviz - ROS Wiki*. (2017). *Wiki.ros.org*. Retrieved 22 October 2017, from http://wiki.ros.org/rviz

[4] *turtlebot_capabilities - ROS Wiki*. (2017). *Wiki.ros.org*. Retrieved 21 October 2017, from http://wiki.ros.org/Robots/TurtleBot

[5] *Converting between ROS images and OpenCV images (C++) - OpenCv Answers: Open Source Q&A Forum*. (2015). *A*nswers.opencv.org. Retrieved 13 July 2017, from

http://wiki.ros.org/cv_bridge/Tutorials/UsingCvBridgeToConvertBetweenROSImagesAndOpenCVImages