UNIVERSITY OF AMSTERDAM

Logbook

February 1, 2020

Student:
Bart de Rooij
Jardenna Mohazzab
UvAnetID
11883138
11701145

Tutor:
Micha de Groot

Lecturer:
Arnoud Visser

Course:
Behaviour Based Robotics

Course code:
5102BEBR6Y

Week 1

Tuesday 07/01/2020

This day we acquired background knowledge about the history of robotics. In addition to the lecture of Arnoud Visser at the UvA, we also looked at the recorded lectures on canvas.

Wednesday 08/01/2020

Today we mainly focused on installing the right software to control the robot. Some python path conflicts arose when installing naoqi and openCV on both of our laptops. Eventually these issues were resolved by not maintaining the prescribed installation method via conda, but installing this by using pip.
After successfully installing all the software we managed to connect to the robot "Jerry" by ip-address. We succeeded in making to make Jerry say "Hello world" by following the instruction of Quest 1.

Thursday 09/01/2020

Today we connected to the robot Phineas.
The first half an hour were spent on indentation errors in the code which were caused by copying code directly from the pdf file. After discovering this problem this was easily fixed.

Square w/o turning:
The walking a square without turning is accomplished by making a proxy in the file globals.py and by using the function "MoveTo". The robot walks straightforward and to the side as you would expect it to do. However, it does drift during walking. Therefore it ends in the same position it started, and thus does not walk a perfect square.

Square with turning:
To achieve a square with turning we used the same moveTo function as before. By setting x and y to zero, but specifying a certain angle, the robot can turn around its own axes. To make the robot turn 90 degrees, we used $\theta = \frac{1}{2}\pi$. We made the function for letting the robot walk a square with turning accept one argument "Distance", which is the distance of the sides of the square.

Circle:
For the circle we used the same moveTo function again. We split it up into two moveTo functions calls where the robot would walk half the circumference of the circle and turn around π radians (half a circle) while doing it.

Problems
A problem that we came across is that the robot drifts. This makes the programming challenging, because you have to keep the drift into account, but this drifting is not constantly.

Week 2

Tuesday 14/01/2020

Today the robot lab server did not function correctly. This meant that we could not work with the robot for most of the time we had. Eventually we were able to connect to the robot and begin with Quest 2. Firstly, we made a snapshot. After that we converted the image to black and white using the inRange function and filtered the image. Lastly, we applied a blur to the picture to smooth it of any white spots.

Wednesday 15/01/2020

Today we started with the Hough circles section of Quest 2. With the imageFilter we built yesterday, it was easy to get this working. Initially we filtered the image three times on the different colour ranges and combined them into one picture, but we later switched over to finding circles per range as to decrease errors. This way it is also possible to keep track of which circle has which colour. A point of improvement is how to handle multiple blobs of the same colour, because now we simply discard those. We tweeked the parameters of the Hough function to find blobs of 15-70 pixels with a minimum distance of 30 pixels between blobs. This was found to give the best results on multiple distances and angles. We managed to finish the calculation functions in the vision file but did not test them intensively yet. At the end of the day we ran into some connection trouble with the robot.

Thursday 16/01/2020

We decided it would increase performance if we would cut out the white paper after a blue blob was found. To implement this cutout function we first tried to use the contours function but this gave problems filling in the whole paper, because it did not only find the contours of the paper but also the contours of the blobs. We decided that the use of a Hough lines would be better. Eventually we managed to get the Hough lines working but the filling in did not work.

Week 3

Monday 20/01/2020

We finished the cutout function. By using the Hough lines on the image we created a mask with the Hough lines. From this mask we would recognize the paper using contours. This contour could then be filled from the centre of the blue blob by using floodfill.

Tuesday 21/01/2020

This morning we started using the sensors. We accomplished that the robot walks a certain small distance, then stops and checks if the sonars are too close to the wall. If not, he walks that same distance again, otherwise the code terminates. Also, we wrote a piece of code so that if the robot detects more than three blobs, it determines the standard deviations from the centres of the blobs. This way if one blob has a greater distance from 3 other blobs, it will be ignored, and the three close blobs will be selected as the right blue and green blob.

Wednesday 23/01/2020

Today we made a function to stand perpendicular to the closest wall, by using the sensors. When the robot is further away from that wall, the same amount of radiance turned will give a bigger difference in the sonar measurement from the left and right side than when it is closer to the wall. That is why we also integrated this into the function.
We also cleaned up the code for the vision module. The code was split up into functions and some improvements were made to the interpretation of found blobs.

Thursday 23/01/2020

This afternoon we tried to let the robot walk through the maze. We tried to create a drift control function, that compensates for the divergence of the legs. So if the robot drifts we did a sonar measurement and let it turn straight to the nearest wall.
A no_wall detector was added as a while loop condition when walking forward so the robot stops in front of a wall.
We also implemented a look around function for when the robot stands in front of a wall. It will take snapshots in several head positions and search for blobs.
We discovered some problems with the vision module where we would get bad results with bad lighting. This was also partially because of our implementation of Hough Lines to cut out the paper in the picture. We want to change the way we do this by using the contour function of openCV which yields a cleaner cutout of the paper.

Moreover we tried to make a main structure to assemble all pieces of code in order to let the robot walk through the maze. However, we noticed that the vision module was not optimal.

Friday 24/01/2020

We worked on the research report, composed research questions and made the general structure of the report and wrote the introduction.

Week 4

Monday 27/01/2020

Restructured and cleaned the vision module. We also made behaviour functions that will help the robot walk through the maze. The functions that we made are look for clues, search three blobs, adjust position, follow clue, drift control, wander around, solve maze.

Tuesday 28/01/2020

We tested out the newly made functions and concluded that they did not work because they were too complex. The rest of the day we worked on simplifying and cutting out complex pieces of code. Eventually we ended up with an algorithm that would make the robot walk straight until finding a wall. To ensure that the robot would walk straight in the corridor, we corrected its course according to the sonar. When a wall was found, we would set it up in front of it to take the picture. Four pictures would be taken and interpreted with a different head position to ensure the highest possibility to find a clue. If a clue was found, the robot would follow it and otherwise it would adjust its position and try again. If no clue can be found the robot will wander around and check left and right and in front of it to find landmarks.

Wednesday 29/01/2020

Today we tested the simplified code. The robot operated significantly better. We also tried to implement a function that made the robot turn its head left and right in order to search for clues. However, this did not work because its shoulder was in the way. After that we let the robot turn left and right after a certain amount of distance walked. The turning of the robot is not accurate, sometimes it turns too much, and other times too little. So in theory our code was correct, in practice however deficiencies of the robot would not allow it to work.