DTU Electrical Engineering
Department of Electrical Engineering

Advanced Autonomous Robots

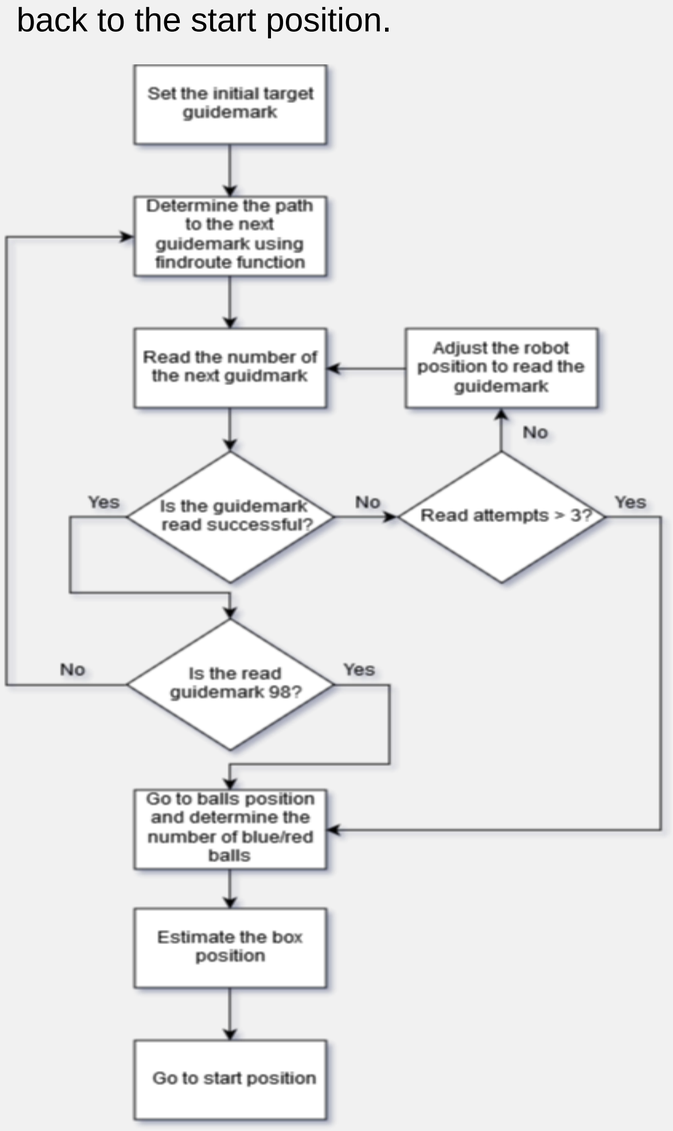
Group 12

Szymon Kowalewski, Javier Fuentes, Omar Gallardo, Jonas Kohlschmidt

Navigation

The Navigation was the most comprehensive part of this project, as it included combining the guidemark reading, the ball detection and the boxfinding. The general approach can be described best in a block diagram showing the different states and associated tasks of the navigation (see picture 1). This approach might be similar to the other groups.

Mentionable is the robustness of our solution. The robot adjusts its position several times when it is not able to read the guidemark. If it still fails to read the next waypoint it will go on with the Ball Detection and the Box Finding before going back to the start position.

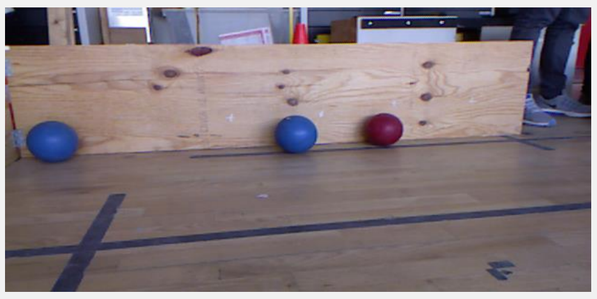


Picture 1: Block diagram of the navigation task

Ball Detection

Positioning to take the picture

In order to take a picture where every possible ball could fit in and to avoid the posibility of being blinded by the box situated in the lower-right corner we made the proper measurements and decided the next moves. The robot would go and face to the guidemark 3 and then tourn 98º towards the balls, where it will take the picture, like we show in the next image (Picture 2).



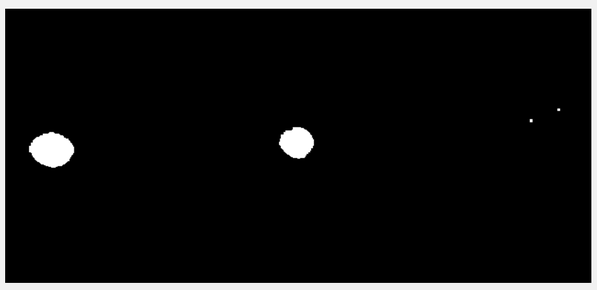
Picture 2: Picture took by the robot to detect the ball

Ball detection plug-in

Once the picture has been taken, it will be saved and then debugged using our C++ plug-in. The way our plug-in works is applying the proper thresholds into a HSV image instead of the YUV. We decided to use this kind of format because it allows you to work with the Hue, the Saturation and the Brightness of the pictures, which is very useful when there are different illuminations. Once the image has been transformed, we apply to both the red and blue balls the min and max values of the thresholds, which have been obtained through experimentation.

Then, we will apply a mask to display all pixels within the chosen thresholds and, in order to make it more robust, we ignore the possible detections that are above 2/3 of the image.

Finally, we utilize the erosion and dilation to remove the small and undesirable detections and, from that final image, is where our plug-in will count the number of blue and red balls.



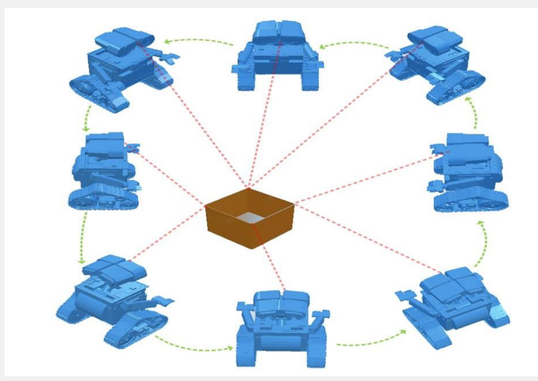
Picture 3: Taken picture after applying the masks to find the blue balls

Box Finding

For the Box Finding part of the challenge we programmed the robot to move around the designated box area and stop at eight locations for scanning the box (see picture 4). After turning towards the box area at each location, the robot localized itself and then performed a laserscan to determine the distances to neighboring objects.

The localization was needed to determine the exact position of the robot and eventually transform this position from the robot to the world coordinate system. The laserscan returns twelve distances to nearby objects, each representing the seperation from robot to objects for a different angle, from -90° to 90° in the robots field of vision. Using the position in world coordinates, the distances and the angles from the laserscan, the location of neighboring objects can be determined.

We excluded all locations outside the box area and extracted the shortest distance from each of the eight scans, which represents the distance from robot to the box. Using this information we could determine the location of the box from eight different positions. The average of these eight box locations finally allowed us to find the real box location with great accuracy.



Picture 4: Represantation of the eight positions the robot used to find the box

Fault Analysis

The first run during the competition was without any problems and we scored the maximum number of points. During the second run an unexpected software exception occurred and the robot stopped during box measurements. We were not able to recreate the failure after the run and additionally the problem has never happened before. Because of that it is extremely hard to determine the reason. One of the possible reasons is that all variables are interpreted as a floating-point variables (doubles). The error occurred when one of the values is compared to another using "==" sign and it is possible that due to lack of precision of a floating-point number we got an unexpected result. We modified the code to avoid that problem in the future.

DTU Electrical Engineering
Department of Electrical Engineering
DTU Electrical Engineering
Department of Electrical Engineering
DTU Electrical Engineering
Department of Electrical Engineering
DTU Electrical Engineering
Department of Electrical Engineering