Second Milestone

The problem

The goal of the milestone 2 was to test the collision avoidance of the milestone 1 and to implement a detection algorithm to detect pucks, poles, and goals from sensor data.

Our approach

Our approach uses only the rgb-pointcloud provided by the kinect sensor because it contains most information. We decided not to use lidar or the 2d rgb image since they do not provide any useful “new” information compared to only the pointcloud. Filtering and editing of the pointcloud was done using the Point Cloud Library. In addition OpenCV was used for detecting goal corners from the floor.

The detection of objects has following steps:

Peparations:

- Listen to robot1/kinect/depth_registered/points to get a pointcloud.

- Transfom received pointcloud to robot1/odom frame using tf.

- Apply voxel filter to cloud to reduce unnecessary data and increase processing speed.

- Edit colors of the pointcloud. In this step all colors that belong yellow/blue/green objects are highlighted to a specific color, for example yellow → rgb = (255,255,0). The highlighting is based on 3D regions in the 3D RGB space.

- Segment cloud into floor and not floor using planar segmentation.

Puck and Pole detection:

- Apply radius outlier removal filter to remove outliers of the not floor pointcloud.

- Use euclidean clustering to divide cloud into region which are separated by space.

- Check all regions if they have the correct size and number of highlighted points

- Add detections to a detection pointcloud which will be published.

Goal detection:

- Set z-value of all points to zero of the floor pointcloud.

- Extract points that are on a color edge (twice for yellow and blue)

- Find corners of the goals by creating a 2D image of the cloud and then doing a hough transformation to find the edges in the image. To get the corners the crossing between lines that are perpendicular are calculated.

- Combine corner detections using euclidean clustering.

- Add detected corners to detection pointcloud which will be published

Combining detections over time (map_node):

- Listen to topic that contains the detection pointcloud and add it a sum cloud that contains also old detections.

- Remove too old detections from the sum cloud.

- Apply radial filter to remove outliers from the sum cloud.

- Use euclidean clustering to combine close object detections to one.

- Publish final cloud that describes the environment.

Currently the algorithm is not very robust against different lighting situations, thus, sometimes error detections occur especially at the detection of the goals since no physical clues are available.

Work division

For the second milestone our team met a couple of times to record rosbags and develop the system. However, most of the code for this milestone was done by Jaakob (>90%).