Institute for Cognitive Systems
Technische Universität München
Prof. Gordon Cheng

Group E: Group E

Concept: Object Recognition and organization

Project description: For this task, NAO is waiting for user input like "please sort these objects". After the speech command, NAO patrols a workspace and search for objects lying on the floor. It is tasked with cleaning up by sorting the objects into storage containers, based on their category. The robot should sort objects in at least three different containers. Objects are selected based on the robots ability to grasp them (e.g. easy to grasp, small objects like a sponge or pingpong ball). Object detection is done via the learned methods in the tutorials, e.g. color or blob detection, or template detection (no aruco markers!). The robot should also be able to avoid obstacles that can not be moved and navigate around them. Obstacles are marked with aruco markers. Once the workspace is cleaned, the robot returns to checking for objects.

Group members: Bruno Feitosa, Aimilios Petroulas, Xinwen Liao

Tasks definition:

A total of $\mathbf{30} \times \mathbf{3} = \mathbf{90}$ **points** will be distributed between the following tasks:

1. **Speech and tactile human-robot interface, ArUco marker detection (5 points).**
   - Speech recognition and generation
   - Tactile interface
   - ArUco marker detection
2. **Autonomous object identification and localization (15 points).**
   - Perform color-based object detection. (5 points)
   - Perform localization of the detected objects by wisely fusing robot kinematics data and the object pixel coordinates while assuming a flat floor. (5 points)
   - Object detection should be carried out after each cleaning cycle (5 points)
3. **Path Planning and Navigation (42 points).**
   - Code a high-level path planning routine allowing to generate an optimal cleaning strategy based on the objects detected by the robot in the room. This function should return a sequence of trajectory milestones that should be followed by the robot between the different detected objects and the storage location. You may use an existing solver for this purpose, such as a traveling salesman solver or a PDDL implementation. Note that you are not required to account for obstacles at this time. (10 points)
   - Code a navigation routine, allowing the robot to move from one trajectory point to the other, using fixed ArUco markers to locate itself in the room. (5 point)
   - Code an online obstacle detection function allowing the robot to detect an obstacles along its path using its cameras and bumpers. The obstacles should consist of a set of at least 3 boxes, labeled by ArUco markers on every visible face. (5 point)

- Code an online obstacle avoidance function based on a simple algorithm (e.g. $A^{\star}$). (10 point)
- Build a reconfigurable map of the robot in its environment (on RVIZ) including the target(s), obstacles and planned trajectory. (9 point)
- Enable fall recovery and self-collision avoidance (using the dedicated Aldebaran API flags). (3 point)

4. **Autonomous grasping (28 points).**

   - Implement a *pre-grasp* behavior. **Hint:** you can manually move the robot to a set of desired poses and measure the corresponding sequence of joint angles via an appropriate ROS topic. You can then simply have the robot re-execute the captured sequence of joint angles... (8 points)
     - Implement an upper-body grasp behaviour in case the object is at torso-level. (5 points)
     - Implement a whole-body grasp behaviour in case the object is on the ground. (5 points)
   - Once in pre-grasp pose, use the Cartesian control API to fine-tune the grasp motion using visual feedback. (15 points)
   - Assess the grasp status autonomously using the camera (5 points)