University of Bremen

German Research Center for Artificial Intelligence

Worksheet 06: Mechanical Design, Actuators, Electronics

Deadline: 01.02.2023
Robot Design Lab
03-IBGA-FI-RDL
WiSe 22-23

Universität Bremen
FB 3 – Mathematik und Informatik
Arbeitsgruppe Robotik
Prof. Dr. Dr. h.c. Frank Kirchner
Dr. rer. nat. Teena Hassan
M. Sc. Mihaela Popescu

1 Quiz on Mechanical Design 20%

Please answer the following questions:

1. The construction process is divided into 3 main parts: specification, principle solutions and design. Please explain what is meant by each of these parts. (6%)
2. What is the main difference between additive and subtractive manufacturing? How do additive and subtractive manufacturing compare in terms of speed, complexity and robustness? (8%)
3. What is biomimetics? How can robotics design benefit from it? Give two examples of a biomimetic design in robotics. (4%)
4. What are the two main design principles used in biomimetics? (2%)

2 Quiz on Actuators 25%

Please answer the following theoretical questions:

1. What is the difference between a stepper motor and a DC motor? (4%)
2. Describe the advantages and disadvantages of brushed and brushless motors.(4%)
3. What is the difference between an inner rotor motor and an outer rotor motor? Name two optimal uses for these two types of motors. (5%)
4. What is a motor characteristic curve? What is it used for? (5%)
5. Give two examples of bio-inspired actuator designs. (2%)
6. What is the working principle of hydraulic cylinders? (5%)

3 Quiz on Electronics 25%

1. Explain the Pulse Width Modulation (PWM) concept and name two practical applications. (3%)
2. Calculate the PWM duty cycle [in %] of a signal with the period $T = 20ms$ and the active time $T_{on} = 4ms$. What is the frequency f of the signal? (3%)
3. Explain the components, working principle and usage of the H-bridge. Moreover, please provide a schematic representation of the current flow in the H-bridge circuit in the two operation modes for motor control. (8%)
4. Describe the following data communication protocols and name an use case for each of them: (6%)
   (a) Serial Peripheral Interface (SPI)
   (b) Universal Asynchronous Receiver-Transmitter (UART)
   (c) Inter-Integrated Circuit (I2C)
5. Consider you want to measure a voltage with an Analog-to-Digital Converter (ADC). You configured your ADC to a resolution of 10bit. The voltage reference for the ADC is a high precision $2.048V$ source. After reading the ADC result, you got a value of $0X00F7$. What is the measured voltage? (5%)
   Hint: First compute the ADC quantization, which is the voltage reference divided by $2^{\text{resolution\_bits}}$. To calculate the measured voltage, convert the hex reading into a decimal number and multiply it by the ADC quantization.

4 Robot Hardware-Software Architecture 20%

In the Robot Design Lab course, we learned about different hardware and software components that are needed to make mobile robots perceive the environment, navigate autonomously and perform complex tasks. Specifically, we looked at the following modules:

- Sensors
- Actuators
- Odometry
- Mapping
- Localization
- Path Planning
- Velocity Control
- Task Planning
- Plan Execution
- Visual Perception

Tasks:

1. Sketch an architecture diagram that connects these components together in a functionally and semantically coherent manner. You can use labeled rectangles to represent each of the above components and arrows to show possible directions of control and data flow between these components. (10%)
2. Explain in detail your reasoning behind this architecture and the way you connected the components. (10%)

5 Perception on Robot 10%

In this task you will run the shape detection nodes from Worksheet 05 directly on the robot! For this, you will first have to copy the `shape_detection` package on your TurtleBot and build it as follows:

1. Copy the `shape_detection` package to the robot via `scp` (replace `xx` with `30+robot_number`):
```
scp -r ~/rdl_ws/src/shape_detection/ ubuntu@192.168.1.xx:/home/ubuntu/turtlebot3_ws/src/
```
2. Connect to the robot and build the `shape_detection` package:
```
ssh ubuntu@192.168.1.xx
cd turtlebot3_ws
colcon build --packages-select shape_detection
```
3. Install the `opencv` dependencies:
```
sudo apt install python3-opencv
sudo apt install ros-foxy-cv-bridge
```
4. Launch the ROS 2 nodes on the robot and start the robot camera node in two different terminal windows:
```
ros2 launch turtlebot3_bringup robot.launch.py
ros2 run image_tools cam2image --ros-args -p height:=720 -p width:=1280
```
5. Run the shape detectors in two different terminal windows on the robot:
```
ros2 run shape_detection circle_detection
ros2 run shape_detection polygon_detection
```

Task:

- Place the robot inside the maze in front of one of the 4 images with circles, squares, triangles or rectangles.
- Run the circle and polygon detection nodes and save the camera images with the marked shapes.
  In order to copy images from the robot to your local PC, you can use `scp`:
```
scp -r ~/turtlebot3_ws/src/shape_detection/results/ ros@YOUR_PC_IP:/home/ros/rdl_ws/
```
- **For every image in the maze, provide two marked images as a result from the circle and polygon detection nodes. Your submission should have in total 8 marked images. (10%)**

6 Perception-Based Autonomous Navigation Ungraded

In this task, you will perform a semi-autonomous version of perception-based navigation within a cardboard maze using the ROS 2 navigation stack, the shape detection nodes and the competition game nodes. This is a preparatory task for the final competition.

Please follow these instructions to start all nodes necessary for this task:

1. On your PC:
   (a) Unzip the ROS 2 package named `competition_game` provided in the supplementary materials of this worksheet.
   (b) Copy the folder to `rdl_ws/src`.
   (c) Build `competition_game` package using `colcon build`:
   `cd ~/rdl_ws`
   `colcon build --packages-select competition_game`
   (d) Launch the ROS 2 navigation stack by following the instructions in Section 2.1 of the `Bonus Worksheet`:
      i. While doing so, replace `map.pgm` and `map.yaml` with the map of the cardboard maze. These files are provided in the supplementary materials of this worksheet in Stud.IP.
   (e) Now, start the following nodes in separate terminals, in the given order:
   ```
   ros2 run competition_game game_controller
   ros2 run competition_game waypoint_finder_tree
   ```
2. On your robot:
   (a) Start the following nodes in separate terminals on the robot, in the given order:
   ```
   ros2 run shape_detection circle_detection
   ros2 run shape_detection polygon_detection
   ```
   (b) Start the camera node on the robot by running the following command:
   `ros2 run image_tools cam2image --ros-args -p height:=720 -p width:=1280`

The game should proceed as follows:

1. Enter the index of the first waypoint to the `game_controller`.
2. Place the robot at the entrance doorway of the maze.
3. Set the initial pose by following the instructions in Section 2.2 of the `Bonus Worksheet`.
4. In `rviz2`, set the 2D goal pose to the location of `Waypoint 1` by following the instructions in Section 2.3 of the `Bonus Worksheet`.
5. The robot should now navigate autonomously to `Waypoint 1` and stop more or less inside the marking labelled `1` on the floor.
6. If the robot does not stop at the desired location, you can set additional goal poses to navigate the robot until it arrives at the desired location. These additional goal poses could also be farther away from the waypoint, so that the robot has enough space to recover from an undesirable pose. Alternatively, you can use teleoperation to make minor adjustments to the robot's pose.
7. After the robot has arrived at `Waypoint 1` and you are convinced with the quality of the camera view:
   (a) Go to the terminal where `game_controller` node is running.
   (b) In this terminal, press the Return or Enter key to confirm that the robot has arrived at the waypoint.
8. Wait for a few seconds to let the behaviour tree process the output from shape detection nodes.
9. After the analysis is complete, the `game_controller` node will print the next waypoint to which the robot should move.

10. Set the 2D goal pose to the location of this new waypoint in `rviz2`.
11. Repeat Steps 4 to 10 until the `game_controller` reports that the game has been completed.
12. Stop the `game_controller` node. Now, when you do `ls`, you should see a log file containing the perception results at each waypoint (except the last waypoint, which is the exit).
13. Inside `rdl_ws/src/competition_game/snapshots/` folder, you will see an image saved at each waypoint (except the last waypoint).

As you can see, you are manually setting the goal poses for the waypoints using `rviz2`. This was needed because our experiments showed some inconsistencies in the fully autonomous navigation through a sequence of predefined waypoint poses. Some tips for troubleshooting:

- In case you face issues with the navigation stack, you could relaunch the navigation stack and set the initial and goal poses accordingly.
- If the above does not solve the issue, you can teleoperate the robot to reach the next waypoint. `rviz` should show you the updated location of the robot.
- If you lost connection to the robot and had to reboot your robot or your PC, you can restart the game from the point where it got interrupted. For this, restart `game_controller` node and enter the index of the next waypoint.
- Remember to empty the folder `turtlebot_ws/src/shape_detection/results` on the robot every time you restart the shape detection nodes.
- If shape detection nodes are not responding, try restarting the camera and/or the shape detection nodes.

7 Competition: Let's Have Some Fun! Ungraded

On our last tutorial (02.02.2023) we will conduct a competition. This is only meant for fun and will not be graded. The setup for the competition will be similar to that of Task 6. However, for the competition we will change the sequence of waypoints and images of shapes. The evaluation criteria are as follows:

1. Correct type of maximum detected shape at each waypoint.
2. Number of detected shapes (circles, squares, rectangles and triangles) at each waypoint.
3. Quality of the image saved at each waypoint.

The above data will be logged by the `game_controller`. After the competition, we will evaluate the group performances based on the logged data and share the rankings to each group individually. Have fun!

8 Feedback

Your feedback is very important to us. Please briefly answer the following questions:

1. How much time did you spend on doing this sheet per person? Anonymize your answer!
2. Was it too easy, easy, ok, hard, too hard?
3. Please tell us what you liked in this exercise sheet.
4. Did you face any difficulties? What should be improved?
5. Any other general remarks?

9 Submission Procedure

- Please use the LaTeX template provided in *StudIP/Wiki* to write your solutions. Upload the PDF file together with source code and other additional materials as a .zip file in StudIP.
- The naming style of your submission should follow the pattern **Gxx_0y_lastname1_lastname2_lastname3.zip**, where *xx* stands for the group number and *y* stands for the exercise sheet number.