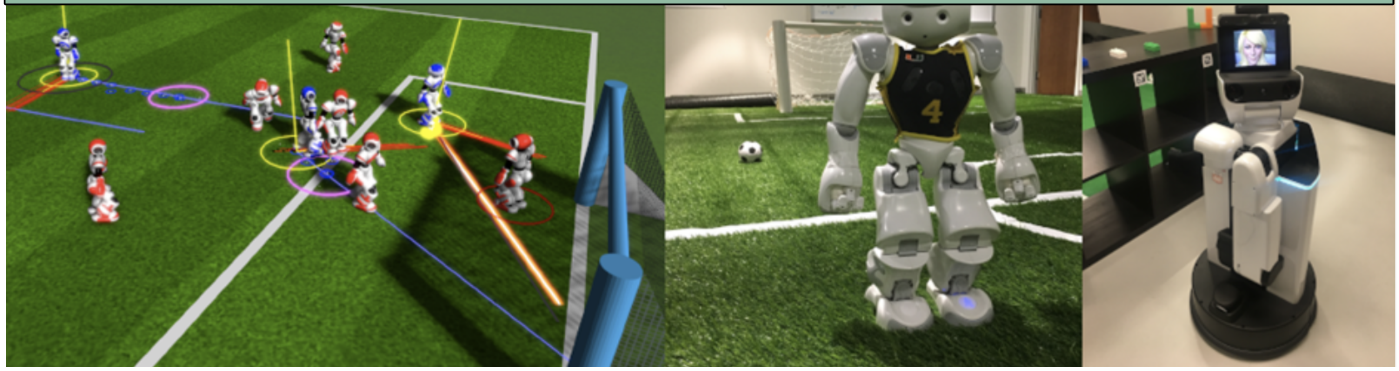
Fall 2024 - CSC398 Autonomous Robots - Assignment 2



Due date: 09/19/2024, 12:30 pm, before class starts. This assignment is worth 20 points.

You will create your first ROS package. You will implement a program that subscribes to a laser scan message from the HSR robot and processes this data. We suggest to refer to the ROS best practices available at `http://wiki.ros.org/BestPractices`. There is a lot of help with regard to conventions, units etc. They also provide a (C++) template you can use as a starting point.

1. `cd` into the `src/` directory off your catkin workspace. `git clone https://classroom.github.com/a/vekyADHc`.
2. Create a package from scratch. Use the command `catkin_create_pkg` to create a new package with the name of your repository (has to be the exact name!) and the dependencies `rospy` (or `roscpp` if you prefer to program in C++), `std_msgs`, and `sensor_msgs`. Refer to the tutorial "Creating a ROS Package" if you are stuck: `http://wiki.ros.org/ROS/Tutorials/CreatingPackage`.
3. Check the `CMakelists.txt` and `package_xml` files.
4. Create a python script in `./src`. Add "`#! /usr/bin/env python3`" in line 1 of the new source file. Add a subscriber to the topic `/hsrb/base_scan`.
5. Add a parameter file with topic name and queue size for the subscriber of the topic `/hsrb/base_scan`.
6. Add a marker (like we did in class) for the visualization of the minimal distance point. You can find more information online: `http://wiki.ros.org/rviz/DisplayTypes/Marker`.
7. Create a callback method for that subscriber which delivers the smallest distance measurement from the vector ranges in the message of the laser scanner to the terminal. The message type is defined as: `http://docs.ros.org/en/melodic/api/sensor_msgs/html/msg/LaserScan.html`. You might want to launch our empty HSR world with `./isaac_sim_hsr_start_empty.sh` and get a distance measure to the green object.
8. Make sure your .py file is executable, i.e. `chmod +x` your python script.
9. Pass the parameter `load` to your node configuration. The file to load is `config.yaml` in the `config` folder (which is in your package home directory).
10. Add a launch file to this package and add your node.
11. Run RViz and visualize the laser scan. Add RViz to your launch file. Create a customized version of RViz for this assignment, which means that the laserscan, robot model, and the markers are loaded. Also, you might need to adapt the size of the laser scan points. You can save your RViz configuration as the default configuration for the future.
12. Add and commit the entire package to github classroom.