COMP.4510/COMP.5490 Mobile Robotics
Fall 2017
Prof. Yanco

# Assignment 3: ROS Race

Out: Thursday, 21 September 2017
Due: Thursday, 28 September 2017, with files submitted before class

***Preparation:***

You should do the tutorials at http://wiki.ros.org/ROS/Tutorials before this race!
You may want to look over the ROS slides form last class as well, particularly for CMakeLists.txt setup.

In your homedir, create a catkin workspace (http://wiki.ros.org/catkin/Tutorials/create_a_workspace).

```
source /opt/ros/rosdesktop_ws/devel_isolated/setup.bash
mkdir –p catkin_ws/src
cd ~/catkin_ws
catkin_make
source devel/setup.bash
```

Download the uml_race starter code:

```
cd src
git clone https://github.com/uml-robotics/uml_race.git
cd ~/catkin_ws/
source devel/setup.bash
catkin_make
```

In the src folder of the workspace you've created (~/catkin_ws/src), create a catkin package (catkin_create_pkg) to contain the node you will be writing.
Name it something containing your name.

```
cd ~/catkin_ws/src
catkin_create_pkg -m [your first name] [your pkg name] roscpp rospy geometry_msgs sensor_msgs std_msgs
cd ~/catkin_ws
source devel/setup.bash
catkin_make
```

For this assignment, you can use either python or C++, but either requires dependencies:
***geometry_msgs*** ***sensor_msgs*** ***std_msgs***

If you use C++, don't forget to catkin_make between test+edit iterations

In each new terminal, you need to source *~/catkin_ws/devel/setup.bash* to use your **catkin workspace.**

**What to do:**

Write a control program to complete the race track simulation provided in the uml_race package. Your robot should travel no faster than 5m/s. This will be enforced by a referee node that is watching you when the robot starts. The starting location of the robot may be anywhere within a 4x10m bounding box around the start location so do not hard code the turns.

The robot has a 5 meter laser scanner that can see 180 degrees in front of it. The output of this topic is published to **/robot/base_scan** and messages are typed ***sensor_msgs/LaserScan***.

The robot listens to commands on **/robot/cmd_velocity**. Messages are of type ***geometry_msgs/Twist***. Twists are composed of an angular Vecto3 and a linear Vector3. Forward velocity is stored in ***linear.x*** while rotational "yaw" velocity is stored in ***angular.z*** .

**To run the racetrack and referee:** roslaunch uml_race racetrack.launch

In the stage window you may see the output of the laser by selecting View > Data. Alternatively, you could *rosrun rviz rviz*, and add a LaserScan display with its topic set appropriately.

It turns out that msg.ranges[0] is the right-hand laser reading. Use msg.ranges[179] to get the left-side reading. You can use any/all of the readings as you see fit to complete the task.

After launching the racetrack, start your node with rosrun in a separate terminal. The racetrack will kill itself after the robot reaches the finish line.

Completion of the race without collision is sufficient. However, if you are inspired by competition, Eric McCann's node runs the track in 49-50 seconds.

***How to Get Help:***

Email jkuczyns@cs.uml.edu if you need assistance on this assignment.

***What To Turn In:***

A zip file containing ONLY your race controller package folder (not the entire workspace).
(from your src folder: *zip –r yourname_race.zip your_race_pkg*)

Copy the zip to your homedir on mercury.cs.uml.edu, then submit it with:

**submit jkuczyns race** *yourname_race.zip*