Activity Sheet: Day 3

Link to bootcamp repo: `https://github.com/UTSARobotics/ros1_bootcamp`

1 Introduction

In this activity, we will create a high-level controller to get a robot to autonomously crash into a pillar!

2 Creating Your Controller Package

2.1 Creation

1. Make package with the following command:

```
catkin_create_pkg smb_highlevel_controller roscpp sensor_msgs
```

2.2 Param server

1. Create a config dir and param file in your custom package

```
mkdir config && touch params.yaml
```

2. Add the following to your params.yaml file:

```
laser_scan_subscriber:
scan_topic: "/scan"
queue_size: 1000
```

3. Make controller program in src folder

```
smb_highlevel_controller_node.cpp
```

4. Program your ‘smb_highlevel_controller_node‘ to:

   (a) Subscribe to scan topic from param server

   (b) Create a publisher on the topic ‘/cmd_vel‘ to be able to send a twist command to SMB. You need to add geometry_msgs as a dependency to your CMakeLists.txt and package.xml:

      i. Update your packages CMAKE.txt. Should include the following lines:

```
    find_package(catkin REQUIRED COMPONENTS
      roscpp
      sensor_msgs
      geometry_msgs
    )
```

      ii. Add the ‘geometry_msgs‘ dependency to your package.xml:

```
<build_depend>geometry_msgs</build_depend>
<exec_depend>geometry_msgs</exec_depend>
```

   (c) Write a simple P controller that drives SMB towards the pillar. Remember to use ROS parameters for your controller gains! Write the code in the callback method of the laser scan topic.

5. Update your packages CMAKE.txt so your new controller node gets executed. Should include the following lines:

```
add_executable(laserscan_min_distance_subscriber src/
   slaserscan_min_distance_subscriber.cpp)

target_link_libraries(laserscan_min_distance_subscriber
   ${catkin_LIBRARIES}
 )
```

6. Go up to your workspace dir and run `catkin_make`. No errors should occur.

7. Test out your program (dont forget to run roscore first)

```
rosrun smb_highlevel_controller laserscan_min_distance_subscriber
   __params:=smb_highlevel_controller/configs/params.yaml
```

3 Launch File

In the launch file you created previously, add lines to load param server and launch node you controller node

1. Update (or create new) launch file in smb_gazebo with:
   (a) To launch pillar.world
   (b) Launch the param server
   (c) Launch your controller node
2. Compile your program

4 RVIZ

1. Launch your updated launch file
2. Open rviz and do the following:
   (a) Change world reference
   (b) Add robot model
   (c) Visualize TF Frames
   (d) View laser scan data