Hackathon tasks

April 5, 2018

1 Align robot parallel to an object

Use the sensors of the robot to position the robot x units away from an object such that the length of the robot is parallel to the object.

Use two nodes, one to run the package and the other that takes a command "go_align" to start moving the robot. Similarly, when a command "stop_align" is given, the robot should stop whatever it is doing(but the node should be running). Once the robot is aligned, a success message should be displayed.

2 Obstacle Avoidance

Use the sensors of the robot to detect an obstacle and move away x units from it.

Use two nodes, one to run the package and the other that takes a command "start_avoid" to start moving the robot. Similarly, when a command "stop_avoid" is given, the robot should stop whatever it is doing(but the node should be running). Once the robot is aligned, a success message should be displayed.

3 Move between nav points

You are to take as input from the user a minimum of 4 nav points(x,y and orientation). Use these nav points to navigate the robot in the order of the points closest to the robot first. Once all the points are reached, a success message is displayed and a new point is taken as user input for the robot to navigate to.

4 Move when an object is detected

Using your webcam/USB camera or you could grab a video off the internet, detect an object of a specific color and move. The detected color needs to be displayed along with an approximate position of where the color is present in the input image/video(left, right, center). When the color is not detected, the robot should stop moving. Keep in mind that, if there is an object present in front of the robot, it should stop moving, keeping a certain distance from the object.

Again, start and stop the running of a node using the commands "start_detection" and "stop_detection".

5 Base Commander

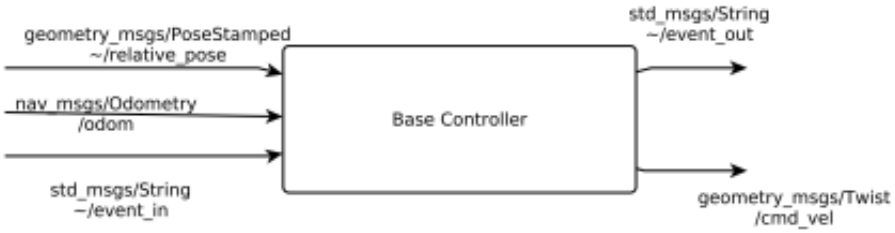

Figure 1

Wait until "e_start" is received. Use the last received input to publish relative poses to achieve required shape or direction. Publish "e_start" to Base Controller to start movement. Wait for feedback from Base Controller for each relative pose before publishing new relative pose. Inputs:

- /event_in: "e_start" and "e_stop" to start and stop the node
- /input: Input strings "left", "right", "circle", "square", "triangle", "forward", "backward"
- /base_status: "e_done" if base has successfully reached relative pose, "e_failed" if base did not reach relative pose.

Params:

- /distance: this parameter defines how many metres to travel if "left", "right", "forward" or "backward" is received as input
- /radius: this parameter defines the radius (in metres) of the circle to if "circle" is received as input
- /side length: this parameter defines the side of the square or equilateral triangle (in metres) that needs to be travelled

Outputs:

- /event out: "e_done" and "e_failed" to indicate success or failure of base movements.

6 Object Following

In this task, you will have to keep an object close to the robot. When the object is moved, the robot should follow the object. Like the previous tasks, use "e_start" and "e_stop" to start and stop the navigation of the robot.

7 Vaccum Cleaner Robot

Using the gazebo world given, find the bumps(dirt) in the map and circle around it to remove it from the goals to execute. In this task, once you visit a dump(dirt), you do not visit it again.

To add the .world file during launch:

- Save the .world file given to you.
- Go to the folder

  `/src/youbot_simulation/youbot_gazebo_worlds/launch`

  and open the file:

  `robocup_at_work_2012.launch`

- Make sure the file is changed to the one below. Ensure that the path of the .world file matches where you have saved it. **Comment the existing code and not delete them**

```
<?xml version="1.0"?>
<launch>

  <arg name="use_sim_time" default="true"/>
  <arg name="gui" default="true"/>
  <arg name="headless" default="false"/>
  <arg name="debug" default="false"/>

  <include file="$(find gazebo_ros)/launch/empty_world.launch">
    <arg name="paused" value="false"/>
    <arg name="use_sim_time" value="true"/>
    <arg name="gui" value="true"/>
    <arg name="headless" value="false"/>
    <arg name="debug" value="false"/>
    <!-- <arg name="world_name" value="worlds/empty.world"/> -->
    <arg name="world_name" value="/home/kv/Downloads/Assignments/FC2/ROS/vaccum.world"/>
  </include>

    <!-- send world urdf to param server
      <param name="world_description" command="$(find xacro)/xacro.py $(find youbot_gazebo_worlds)/urdf/robocup_at_work_2012.urdf" />

    <!-- spawn uploaded world model
      <node pkg="gazebo_ros" type="spawn_model" name="gazebo_world_model" args="-urdf -param world_description -model world -x 0.0 -y
      respawn="false" output="screen" />
  -->

</launch>
```

Figure 2

- `catkin_make`
- `source ~/.bashrc`
- `roslaunch youbot_gazebo_robot youbot_base_only.launch`
- Just like the other tasks, make sure that you make provision to start and stop execution using a command "e_start" and "e_stop".