Exercise and new Assignment!

Carmine Tommaso Recchiuto, Phd

Exercise

- Spawn 2 robots corresponding to the robot description robot2.xacro
- robot 1 should be controlled by a node that gives random linear and angular velocities
- robot 2 should be controlled by a node which implements an algorithm for tracking robot1, only relying on the camera

Implementation

- Launch file using namespaces
  - ➢ N.B. In this case, namespaces should not be used in the robot description

- Two new nodes developed:
  - ➢ random_robot
  - ➢ robot_following.py

- Robot spawned in the empty world

- roslaunch bridge_example gazebo_ex.launch

Implementation

- random_robot:
  - It periodically updates the linear and angular velocities (on the cmd_vel topic) by giving random values between -0.3 and 0.3 for the linear velocity and -0.2 and 0.2 for the angular velocity.

- robot_following.py:
  - It slightly modifies the example_ball.py shown in the video! It just add a controller for the linear velocity, which is now inversely proportional to the difference between the radius of the detected robot and a fixed value (image-based visual servoing). Also, the mask has been adapted to the robot's colour (orange). Please consider that in ROS newer versions, it may be necessary to change the sign of the angular velocity controller.

II assignment - starting

You may download some starting files for performing the second assignment from the git repo CarmineD8/exp_assignment2

- git clone https://github.com/CarmineD8/exp_assignment2.git

Inside this package you will find:

- the world folder, with the world used for implementing the simulation
- the urdf folder, with the description of a robot model, a human and a (big) ball
- the scripts folder, with a python ROS node which implement an action server to move the ball around
- the action folder, with the definition of a custom action message
- the launch folder, with a launch file for executing the simulation

II assignment - starting

In the launch file:

- we start the (gazebo) simulation with a custom-built world, where a 8 x 8 meters area is defined
- In this word, we spawn:
    - a green ball
    - a human
    - a robot (the model used during the previous exercises)

- The ball is defined as a robot with one link and the *planar_move* plugin, which allows for moving the ball in the environment by using the cmd_vel topic. Please consider that the ball has been defined with no collision elements (and also zero gravity, otherwise it would have fallen through the ground!)

II assignment - starting

Finally, the node go_to_point_ball.py has been prepared: it is an action server that moves the ball in the *goal* position (x,y,z)

You can start the simulation with roslaunch exp_assignment2 gazebo_world.launch, and move the ball by publishing on the topic *reaching_goal/goal*

II assignment - requirements

- The robot should become a wheeled dog robot, which means that is should have:
  - A neck (fixed joint)
  - A head with at least one actuated degree of freedom (yaw) and a camera on top

- It may just be a cylinder for the neck and a box for the head, but of course you are free to use the structure that you prefer and to free your imagination.

II assignment - requirements

- The wheeled dog has three behaviors:
    - Sleep
    - Play
    - Normal

- The human can interact by:
    - Moving the ball around (by giving a command position to the ball, then waiting for a number of seconds, and giving another command position to the ball)
    - Making the ball disappear (send a position command with a negative z)

II assignment - requirements

- The sleep and the normal behavior are exactly the same implemented in the first assignment (however now they should be implemented with a “real simulated robot”)

- So in the sleep behavior, the robot reaches a predefined location in the arena, it stays there for some times, and when it wakes up it goes in the normal behavior

- In the normal behavior the robot moves randomly in different points of the arena

II assignment - requirements

- The robot goes from the normal behavior to the play behavior when he sees the ball in the environment. In the play behavior:
    - it starts following the ball;
    - when the ball stops (n.b. the node controlling the robot cannot subscribe to topics related to the ball. So this means that the robot, tracking the ball, stops) it moves the head on the left of 45 degrees, it keeps the head in that position for a number of seconds, then it moves the head on the right, it keeps the head on the right for a number of seconds, then again on the center.
    - Once moved the head, it keeps tracking the ball until it stops again.
- The robot goes back in the normal behavior when it cannot find the ball for a certain amount of time

II assignment - objective

- Build a robot model with additional links and joints.
- Build a ROS architecture to implement the robot’s behaviors and simulate it on Gazebo.
- Provide a launch file to start all nodes used in the simulation

II assignment - objective

- You may obviously start from the assignment I, and build on top of it. You will just need to modify a little the finite state machine and the architecture

- Also in this case you may use randomness to stress the system and evaluate its robustness

II assignment - submission

- Send a link to a github repository to:
  carmine.recchiuto@dibris.unige.it and luca.buoncompagni@edu.unige.it

- The repository should contain:
  - All the developed code
  - Documentation with Doxygen, docstrings, or similar
  - A README.md file with the report of your work

Deadline: 18th December

II assignment – README template

- Brief introduction (couple of sentences)
- Software architecture and states diagrams (a paragraph of description each, plus a list describing ROS messages and parameters)
- Packages and file list (to navigate in the repository based on 2)
- Installation and running procedure (including all the steps to display the robot’s behavior)
- System’s features (1 or 2 paragraph)
- System’s limitations (1 or 2 paragraph)
- Possible technical improvements (1 or 2 paragraph)
- Authors and contacts (at least the e-mail)

Deadline: 18th December

II assignment – README template

We mainly evaluate:

- the design of the software architecture included: communication paradigms, interfaces and parameters
- the quality of the code and documentation
- the capacity of designing robot descriptions in the correct formalism
- the ability to test the architecture and evaluate its outcomes, i.e. behavior consistency
- the quality of the repository and README file