Instructions

October 21, 2019

Introduction

This repository has been created to test the skills of potential recruits regarding the software tools used at Forssea Robotics: ROS, git, C++, python...

The candidates must complete a few exercises in a given time, and present their results at a given date.

One should remember that mastering the tools named above and finishing all the exercises is not an absolute necessity, and that being able to expose the problems one couldn't solve is also important. Therefore, both technical and communication skills will be considered.

1 Exercises

The candidate must have a working ROS installation[1] (on an Ubuntu-like OS). Using Windows is also possible, but not recommended. The candidate should stick to the ROS coding guidelines[2] [3].

1.1 Preliminary steps

This repository contains a ROS package, that contains a ROS node (called *turn_ counter*) written in python. It has 2 separate branches: master and develop.

The candidate must fork the repository [4], and work on his/her fork. Once he/she is done with the exercises, he/she must create a pull request to the develop branch of the Forssea's repository.

You should commit your work after finishing each exercise.

1.2 Readings (~ 15 min)

Read the code provided, and explain what *turn_ counter* does in a few words.

Hints:

One can use internet, and especially the following websites:

- ROS tutorials[5] and documentation[6]
- Wikipedia for math related concepts[7] [8]

[1] http://wiki.ros.org/melodic/Installation/Ubuntu
[2] http://wiki.ros.org/CppStyleGuide
[3] http://wiki.ros.org/PyStyleGuide
[4] https://help.github.com/en/articles/fork-a-repo
[5] http://wiki.ros.org/ROS/Tutorials
[6] http://wiki.ros.org/
[7] https://en.wikipedia.org/wiki/Integral
[8] https://en.wikipedia.org/wiki/Quaternion

1.3 Testing (~ 30 min)

The goal here is to create a ROS Node one can use to test that *turn_ counter* is working properly.

1. Create a C++ ROS Node in the package provided doing the following:
   - Simulate an IMU placed on a robot moving in circles (only consider a robot moving on a 2D plane)
   - Send the IMU message (using basic ROS IMU message type) on a topic
2. Create a launchfile running both nodes correctly connected, to test the node provided in the package
3. Place it in a launch folder in the package
4. Don't forget to adjust the CMakeLists.txt and the package.xml files with your dependencies, and to test your code.

Hints: The robot should be simulated using the following equation:

$$\begin{aligned} x_{k+1} &= x_k + 0.1 \cdot \cos(\theta_k) \\ y_{k+1} &= y_k + 0.1 \cdot \sin(\theta_k) \\ \theta_{k+1} &= 0.05 \cdot k \end{aligned}$$

where x_k is the easting of the simulated robot, y_k is its northing and θ_k is its heading at time step k. You can use θ_k to fill the IMU message.

The following websites may be of use:

- http://docs.ros.org/melodic/api/sensor_msgs/html/msg/Imu.html
- http://wiki.ros.org/ROS/Tutorials/WritingPublisherSubscriber%28c%2B%2B%29

1.4 Sharing your work (~5 min)

Create a pull request as indicated earlier. Don't forget to add explanations about the new features you included in the package.

1.5 Automatic testing (~ 10 min)

Explain how you would set up a Continuous Integration tool to test that *turn_ counter* and your node are interacting and working properly.

1.6 Presentation (= 15 min)

Present your work using a PowerPoint/PDF document.

Notes

- Do not hesitate to ask for help, especially when it comes to installing ROS on your computer or understanding the maths in exercise 1.3.
- The indicated duration for each exercise corresponds to how long it would take us to do the job.