C++ coding evaluation - teach-repeat

This evaluation contains a coding challenge aimed at measuring your C++ programming and design making on robotics-related algorithms skills.

There is no time limit to complete this evaluation. Keep it simple - we anticipate you spend about 6h on this task if you have a recent Ubuntu version installed. Your solution will be discussed as part of the technical interview that will follow. You can use the Internet, useful third-party libraries, or other resources to help you answer these questions, but please don't ask others for help. Your code must compile in a Linux environment (Ubuntu preferred) with modern C++ standards. Please do not write in C.

Next to the functional aspects we would also value application of best practices for non-functional aspects, like consistent code style, a reasonable level of documentation, testability and run-time performance.

Please don't distribute this evaluation, as we also give it to other aspiring candidates.

Thanks for your interest in Intermodalics!

Task

For a mobile indoor robot manufacturer you are tasked to create a teach-repeat navigation system:

- The user is an engineer that knows the Linux command line.
- As a user,
  I want to teach to the robot a path by teleoperating it (joystick),
  so that I can later (after a potential reboot of the robot) command the robot to drive the path autonomously.
- As a user,
  I want to be able to visualize the path in 2D,
  So that I can check the taught-in path
- As a user,
  I want to be able to command the same robot to execute one of many taught paths,
  Such that the robot as accurately as possible follows my taught path while I can do something else in the mean-time.
- *Focus on the teach-and-repeat part*

    - *No obstacles, only driving forward, taught path is coming back to start point*
    - *Robot will start standing still already on the starting point and orientation*
    - *Drive the taught path as accurately as possible*
- *Create the more simple solution that fits the requirements, mention in comments what loose ends or special cases to handle*
    - *Prefer 'easy to implement' over 'the best choice', you can mention the better/best choice where relevant*
    - *Mind readability for the reviewer*

Target platform

- A TurtleBot3 Waffle Pi
    - Simulation, visualization and support software available on
        - https://emanual.robotis.com/docs/en/platform/turtlebot3/simulation/#gazebo-simulation
        - https://emanual.robotis.com/docs/en/platform/turtlebot3/quick-start/#pc-setup
    - You can treat the localization (available on /tf) and low level robot control (consuming /cmd_vel) as given and start from there.

Deliverables

- Architectural overview of the application to communicate your intention to your colleague developers.
- Software implementing a functional prototype (simulation, to be shown during the interview), indicating in comments where more advanced approaches you would consider for the given scope.
- The teach and repeat part are to be implemented yourself. You can make use of support libraries like Eigen.
- Mainly C++ and ROS2 on Linux
- A PR on a private github repository (give access to the reviewer). Put the architectural overview in the PR description.