Homework 3

Robotics 2024 - ShanghaiTech University

In this homework you'll properly evaluate the accuracy of your odometry from HW2 using the ground truth from the tracking system (Task 1). Then you'll use the popular 2D SLAM algorithm gmapping to create 2D maps from the 3D LiDAR scans of the figure 8 trajectory (Task 2). Finally you'll program a scan matching algorithm based on ICP (Task 3).

1 Evaluation (35 %)

When collecting the figure 8 dataset we actually were also collecting IMU data, 3D LiDAR data and ground truth tracking data from the tracking system installed in the robotics lab. Download and unzip the following files to get rosbags we need for this homework:

- https://robotics.shanghaitech.edu.cn/static/robotics2024/figure8_sensor.zip
- https://robotics.shanghaitech.edu.cn/static/robotics2024/figure8_tracking.zip

First lets take a look at the tracking data. It has tf data as well as `geometry_msgs/PoseStamped` messages. To visualize the pose data in rviz follow these steps:

- You'll need to replay with the argument `--clock-topics-all` to use the simulated time from the rosbag and add `use_sim_time:=true` to all your `ros2 run` commands.
- You need to create a tfree with frames "world" and "marker":
  `ros2 run tf2_ros static_transform_publisher 1 0 0 0 0 0 1 world marker`
- Select "world" as fixed frame in rviz.
- Add a "Pose" display to view the topic `"/motive_marker/pose"`
- Note that the tracking system uses the y-axis for "up" - so the motion of the tracked path will be in a different coordinate system. Quick fix: display the Grid in the "XZ" Plane.
- Additionally note that the frame of the tracking marker w.r.t. the tracking system is not the same as the frame of the robot w.r.t. the initial robot pose (I.e. the marker frame does not point in the robot forward direction).

You will need to record your estimated odometry into a rosbag for evaluation. Remember to add parameter `--use-sim-time` to `ros2 bag record`.

Please check this python package for evaluation of odometry and SLAM: https://michaelgrupp.github.io/evo/

Use the tool to create a report (HW3report.pdf) using LaTeXwhere you evaluate the accuracy of:

- Your HW2 solution
- Your HW3 Task 2
- Your HW3 Task 3

The report should include the according diagrams as well as a table with the comparison. There should also be a conclusion where you compare the results and **make suggestions for further improvements**.

You may send your report to Prof. at least one day before the deadline to get feedback on your report and a chance to improve it.

In your HW repo create a directory HW3 and there a sub-directory "report", in which you add the source files as well as the report.pdf.

2 GMapping (20 %)

GMapping is a classical and popular 2D mapping algorithm. It is a particle-filter based SLAM approach for 2D LiDAR data, developed since 2005 or earlier. Take a look at the according paper: https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=4084563.

Your task is to run gmapping with the figure 8 dataset and extract the path as text file for use in Task 1. Here is a ROS2 wrapper of gmapping https://github.com/Project-MANAS/slam_gmapping. See http://wiki.ros.org/gmapping on how to run gmapping. Since our LiDAR is 3D you'll need to turn the 3D data into a 2D laserscan, e.g. by using this package: https://index.ros.org/p/pointcloud_to_laserscan/#iron

You will need to record pose from gmapping into a rosbag for evaluation in task 1. Remember to add parameter `--use-sim-time` to `ros2 bag record`.

In the HW3 directory create a folder "task2" in which you put a launchfile `hw3_gmapping.launch` that does the following:

- Replays the figure8 bagfile (if needed with a lower speed, e.g. –rate 0.1)
- Runs gmapping with the appropriate parameters
- Runs `pointcloud_to_laserscan`
- Starts rviz WITH the proper .rviz file (you can save/ load a layout/ data display setup file File– >Save Config As and also specify it on the command line/ in a launch file!)
- Starts the program that dumps the pose/ path info from gmapping into a text file for use in the evaluation

3 ICP Scan Matching (45 %)

In this task you will implement a node from scratch to do 3D scan matching via 3D ICP. You do not have to implement ICP yourself. We suggest you use the ICP methods from the point cloud library: https://pointclouds.org/documentation/group__registration.html https://index.ros.org/p/pcl_ros/#iron. We suggest you use C++.

You should implement a well working ICP scan matching approach and test it with the figure8 dataset. Output should be 3D global pose as a poseStamped message as well as the text file for use in Task 1.

It is up to you to determine how you want to pre-process the LiDAR data. Possible candidates are: cropping the point cloud; removing noise; downsampling; removing the ground plane; etc. You can utilize all the tools that pcl is providing for that. You may also utilize some initial guess (but not the tracking system of course), e.g. use constant velocity; HW2 odometry; IMU data; or a combination of the former (but it has to be implemented by you - no additional external libraries other than pcl, Eigen and the core ROS stuff).

In the HW3 directory create a folder/ ros package `task3_icp` in which you put a launchfile `hw3_icp.launch` that does the following:

- Replays the figure8 bagfile (if needed with a lower speed, e.g. –rate 0.1)
- Runs your ICP scan matching program
- Starts rviz WITH the proper .rviz file (you can save/ load a layout/ data display setup file File– >Save Config As and also specify it on the command line/ in a launch file!)

`task3_icp` should also obviously contain all files needed for the package/ needed to compile it (in their appropriate folders).

Also create a video of your task 3 ICP in action: I want to see the robot pose updating, the point cloud updating its pose (so publish proper /tf), the static frame should be the map frame/ world frame that you choose. I also want to see a path (your icp node should publish it), similar to the one in HW2. Make a video of that and put it in the HW3 directory called `icp.mp4` with a **maximum size of 50MB**!

4 Submission

Your submission in your gitlab homework repo consists of the following files:

- `HW3/report/report.pdf`
- all the latex sources for that report in `HW3/report`
- `HW3/task2/hw3_gmapping.launch`
- The ros node `HW3/task3_icp` with all the source files
- `HW3/task3_icp/hw3_icp.launch`
- `HW3/task3_icp/icp.mp4`