FIRST ROBOTICS PROJECT

ROBOTICS

POLITECNICO
MILANO 1863

THE PROBLEM

Provided data:

- Odometry from the robot
- GPS data
- PointCloud

DATA

Format: ROS Bag file

play the bag with the command:

rosbag play --clock robotics.bag

Data:

- /fix gps data
- /odom odometry from encoders
- /os_cloud_node/points pointcloud from lidar

THE PROJECT (First Node)

- Create a ROS package called ***first_project***
- Create a ROS node to compute the odometry from gps data:
  - write a node called ***gps_to_odom*** which publish:
    - odometry message:
      - type: ***nav_msgs/Odometry***
      - topic name: ***gps_odom***

THE PROJECT (First Node)

- to convert gps data to odometry:
  - convert (latitude-longitude-altitude) to Cartesian ECEF
  - convert Cartesian ECEF to ENU
  - ENU is a relative position, so you need to specify the reference point
- gps_to_odom should have three parameters :
  - lat_r
  - lon_r
  - alt_r
- These parameters should be set in a launch file, for this project you can set manually to the first value from GPS:

THE PROJECT (First Node)

- GPS gives you only position
- For the odometry you also want orientation
- In this scenario we work on a 2D plane, so we want the robot heading
- After computing the robot position in ENU you can use consecutive poses to estimate the robot heading

THE PROJECT (Second Node)

- Inside the ***first_project*** package
- Create a ROS node to convert from odometry to tf:
  - write a node called ***odom_to_tf*** which subscribe to:
    - odometry message:
      - type: ***nav_msgs/Odometry***
      - topic name: ***input_odom***
  - publish with a tf broadcaster an odometry between two values set as node parameters:
    - the two node parameters are called *root_frame* and *child_frame*
    - the node is supposed to be started from launch file with topic remapping for input and parameter for the tf_broardcaster

THE PROJECT (Second Node)

- You should write a launch file which create two instances of this node, one to publish as a tf the odometry from the encoders, and one the odometry from gps
- both published tf should have the same root set as world, but the child frame should be *wheel_odom* and *gps_odom* respectively

THE PROJECT (Third Node)

- Inside the ***first_project*** package:
  - Final node for lidar data visualization
  - The node subscribe to the /os_cloud_node/points topic and change the frame set in the header
  - The value set in the header is regulated by a dynamic reconfigure callback which allow to dynamically change it to be set to the *wheel_odom* or *gps_odom* frame
  - The final node should allow the user to select from rqt_reconfigure to which tf the lidar is connected

THE PROJECT (Launch file)

- The three node should all start from a single launch file called **launch.launch**
- The launch file should also open rviz
- the only command to start the project should be:
  roslaunch first_project launch.launch

THE PROJECT (debugging)

- If the odometry and the tf are set properly you should see in rviz the laser scan data and the odometry
- If you set world as root frame in rviz you should see the laser scan data features stay almost still, and update while the robot moves in the environment
- If the lidar data moves around probably the odometry or the tf are wrong
- Also, check the odometry value, the gps and wheel odometry should have similar values, and diverge a bit trough time

THE FILES

https://goo.gl/GonArW

First_Project folder

Deadlines and requested files

-Send **only** a tar.gz file

-Send via e-mail both to Simone Mentasti and Matteo Matteucci

-name the e-mail “FIRST ROBOTICS PROJECT 2024”

-Inside the archive:

- info.txt file (details next slide)
- folders of the nodes you created (with inside CmakeLists.txt, package.xml, etc...)
- **do not send** the entire environment (with build and devel folders)
- **do not send** the bag files

Deadlines and requested files

File txt must contain only the group names with this structure

codice persona;name;surname

You can add another file called readme.txt with additional info. I will not always look for it. But if something goes wrong I'll check for explanations.

Some more requests

Name the archive with your codice persona

Don’t use absolute path

The project need to be written using c/c++

Deadlines and requested files

Deadline: 3 May (1 month)

Max 3 student for team

Questions:

-write to me via mail (simone.mentasti@polimi.it)

- do not write only to Prof. Matteucci

Formulas

Latitude-longitude to ECEF:

$$X = (N(\phi) + h)\cos\phi\cos\lambda$$
$$Y = (N(\phi) + h)\cos\phi\sin\lambda$$
$$Z = \left(N(\phi)(1 - e^2) + h\right)\sin\phi$$

Where Φ is latitude, λ is longitude, *h* is altitude, and

$$N(\phi) = \frac{a}{\sqrt{1 - e^2\sin^2\phi}},$$

Formulas

ECEF to ENU:

You need both robot position and reference position (lat_r, lon_r, alt_r) in ECEF Coordinates

Then you can apply the formula:

$$\begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} -\sin\lambda_r & \cos\lambda_r & 0 \\ -\sin\phi_r\cos\lambda_r & -\sin\phi_r\sin\lambda_r & \cos\phi_r \\ \cos\phi_r\cos\lambda_r & \cos\phi_r\sin\lambda_r & \sin\phi_r \end{bmatrix} \begin{bmatrix} X_p - X_r \\ Y_p - Y_r \\ Z_p - Z_r \end{bmatrix}$$

Where Φ_r is latitude, λ_r is longitude of the reference point (n.b., you are using sin and cos, so make sure they are in radians/degree, depending on the library you use)

Changelog

- conversion: now is from LLA to ENU, previously was LLA to NED
- formula for LLA to ECEF and ECEF to ENU
- updated naming and parameters for consistency with formulas
- new bag file and updated instructions. Now the bag file contains 3D data instead of 2D for better visualization