CS-475 Assignment 1
Kalman Filter

Kalykakis Emmanouil
csdp1210@csd.uoc.gr

Release date: Sunday 05/03/2023
Deadline: Monday 13/03/2023, 23:59

1 Introduction

In this assignment, we are going to help our turtlebot to localize itself by using the Kalman Filter (**KF**). The Kalman filter allows us to combine a variety of potentially erroneous/inaccurate sources and obtain an estimate about the state of our robot that is more accurate than any individual source. Our robot has 2 sources of positional data:

1. It is equipped with a GPS sensor which returns the position of the robot (x,y).
2. It tracks the rotation of each of its wheels and returns how much the robot has turned.

In a perfect world, we would just need one of the aforementioned sensors in order to perform successful localization. In practice the wheels are subject to slip which may cause the robot to believe it's turned a different amount than it actually has; this error accumulates over time (aka "drift"). While GPS simply is not accurate enough (its resolution isn't high enough) on its own for the tasks we want our robot to do. For the purpose of this assignment we'll assume that all sensing uncertainties can be modeled in terms of Gaussian white noise and therefore the KF can be utilized to filter data streams and obtain more reliable estimates about the robot's position.

2 Setup

Open a terminal (keyboard shortcut: Ctrl+Alt+T) to navigate to your home directory and then type:

```
mkrir -p ~/asgmt1_ws/src
cd ~/asgmt1_ws/src
```

If you ever want to open a new tab in an existing terminal session, use the keyboard shortcut Ctrl+Shift+T. Type the following commands in order to clone the Turtlebot3 packages from the repository to your workspace

```
git clone https://github.com/ROBOTIS-GIT/turtlebot3_simulations.git
git clone https://github.com/ROBOTIS-GIT/turtlebot3.git
```

Also, if you haven't already installed the following package, type the following command:

```
sudo apt-get install -y ros-rosdistro-turtlebot3-msgs
```

Replace rosdistro with the name of the installed ROS distribution in your PC. You can find this, by typing the command: `echo $ROS_DISTRO`
Copy the directory `cs475_asgmt1` with its contents into the directory:

```
/home/<user_name>/asgmt1_ws/src
```

In order your system to see the new ROS workspace and the necessary files for the Turtlebot3 robot, you have to source the `devel/setup.bash` file every time by using the command:

```
source /home/$(whoami)/asgmt1_ws/devel/setup.bash
```

and exporting the variable `TURTLEBOT3_MODEL` with the command:

```
export TURTLEBOT3_MODEL=burger
```

I suggest to copy the given file `run.sh` into the folder of the workspace (`asgmt1_ws`) in order to avoid every time you open a new terminal session to execute the previous two steps. The only thing needed is to set execute permissions to it using the command: (`chmod u+x run.sh`). Type the following commands in order to compile the packages:

```
cd ~/asgmt1_ws/
catkin_make
```

You can start and load everything when you type the command `./run.sh` (don't forget to type the dot character at the beginning of the command).

Everything is ready for you regarding the package "setup", you don't need to change anything in the CMakeLists.txt or package.xml. Inside this package you will find the launch/ folder, which contains the launch file (burger.launch). This file provides a convenient way to start up multiple nodes and a master(`roscore`), as well as other initialization requirements such as setting parameters. The ONLY thing that you will need to modify is the src/kalman.py file. (Although playing around and tweaking stuff is highly recommended, always keep backup and test your code on the given template package.) Finally, don't change the name of your node from "kalman.py".

3 Implementation

For this assignment you will implement the KF by filling up the `???` inside the src/kalman.py. Practically you will need to code the mathematics from slide 17 of Bayes Filter Implementations presentation (2.kalman.ppt).

As mentioned before you will have to fuse two sensors: GPS and motion sensor in order to get a better estimation about the robot's position (x,y) and orientation (dir). Your noisy inputs come from the "/odom" topic and they named: `gps_x,gps_y,deltaDir`. deltaDir can be considered as your control input i.e. how much the robot is commanded to turn at each timestep. I suggest to comment blocks of your code and take it step by step, checking every time if the dimensions of your matrices are correct and if they contain the expected values.

Simulation

This time, instead of Gazebo (running on the background) you are going to use the Rviz. Rviz is a powerful tool, built-in for ROS and helps with the visualization of data. You can start the simulation by opening a terminal window and type:

```
roslaunch cs475_asgmt1 burger.launch
```

Then, an Rviz window (displayed at Figure 1) should appear and the robot will start moving forever with **constant linear speed=0.7** and **anglular velocity =0.7**. When you have completed your implementation, Rviz will subscribe to the "/kalman" topic and visualize the state of your robot. Start up the kalman node with the command:

```
rosrun cs475_asgmt1 kalamn.py
```

If you want this node to start automatically along with all the rest, you can uncomment the second to last line of the `burger.launch` file.

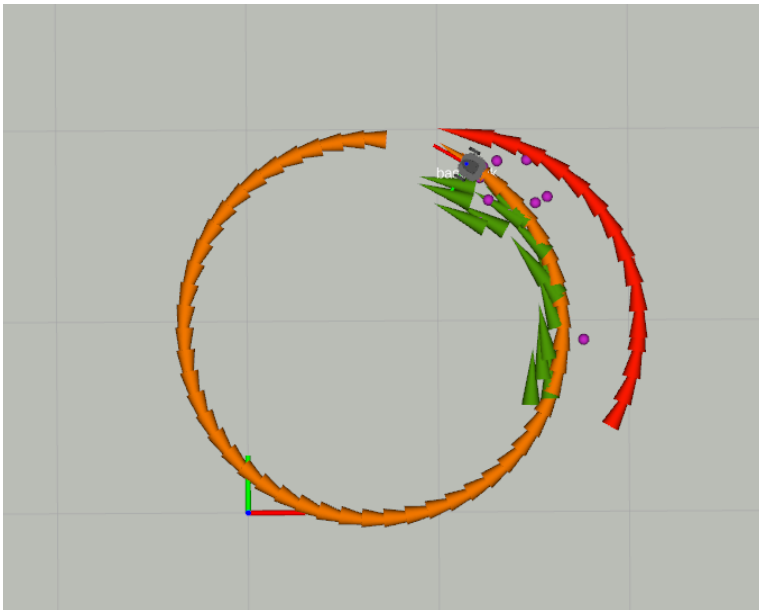

Figure 1: Screenshot taken from Rviz

In the above image, the green arrows represent the last 15 Kalman filter estimates, the purple dots the noisy GPS measurements and the red arrows the noisy odometry. The orange arrows represent the mostly accurate position and orientation of the vehicle.

4 Submission

Send your node (`kalman.py`) attached via email at: **csdp1210@csd.uoc.gr** Don't forget to add at the top of the file your name and your registration number as a comment. The subject of the email should be the following: **[CS475]: Assignment 1 submission**. The deadline is due to **Monday 13/03/2023 23:59**