深 圳 大 学 实 验 报 告

课程名称：机器人学导论

实验项目名称：**Localization and Mapping**

学院：电子与信息工程学院

专业：电子信息工程

指导教师：郑琪

报告人：陈闻天 学号：**2023280259**

班级：**04**

实验时间：**2024** 年 **10** 月 **15** 日

实验报告提交时间：**2024** 年 **10** 月 **21** 日

教务处制

Aim of Experiment:

1. Master the principle of Kalman Filter and apply it in experiment.
2. Learn the principle of localizing Robots in Landmark Map.

Experiment Content:

1. Complete the code to implement the observation of temperature indoor.
2. Simulate the Dead Reckoning using Odometry and the Landmark Maps.

Experiment Process：

1. Complete the code:
   a) From the provided reference code, learn that X, Z, G, P, Xkf represent respectively real value, observation value, Kalman Gain, covariance and Kalman optimal estimate.
   b) Based on Kalman Filter, complete the code to implement task.
2. 6.1.1 Modeling the Robot:
   a) Create a odometry covariance which is used to initialize model for robot.
   b) Set the speed and steered-wheel angle of robot and run it much times.
3. 6.1.2 Estimating Pose:
   a) Set the initial covariance which is used to create EKF object ac argument.
   b) Run the EKF object 400 times and plot the outcome in the

windows.

4. 6.2.1 Localizing in a Landmark Map:
    a) Create a specified map with 20 landmarks.
    b) Create a sensor class with a specified uncertainty covariance.
    c) Create a robot model based on BicycleVehicle.
    d) Localize the robot while it is moving, with Landmark Map.
5. 6.2.2 Creating a Landmark Map:
    a) Create a specified map with 20 landmarks.
    b) Create a EKF object with a specified covariance.
    c) Run the robot 1000 times and plot the outcome of the Run.
6. 6.2.3 EKF SLAM:
    a) Initialize the basic settings, including map, covariance and some models and objects used before.
    b) Run the EKF 500times and plot the map.
    c) Transform map to SE(2).

1Data Logging and Processing:

1. The completed code as shown in Figure 1:

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% 模拟房间温度和测量过程，并滤波
for k=2:N
% 第一步：随时间推移，房间真实温度波动变化
X(k)=X(k-1)+W(k);
% 第二步：随时间推移，获取实时数据
Z(k)=H*X(k)+V(k);
% 第三步：Kalman滤波
G=(P(k-1)*(H'))/(H*P(k-1)*(H')+R);
Xkf(k)=Xkf(k-1)+G*(Z(k)-F*Xkf(k-1));
P(k)=P(k-1)-G*H*P(k-1);
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

Figure 1

2. 6.1.1 Modeling the Robot

   a) Aim of experiment: model the motion of robot with the odometry noise.

   b) Code and comment:

6.1.1 Modeling the Robot

```
%create V which is odometry covariance
V = diag([0.02 deg2rad(0.5)].^2);
%create BicycleVehicle object to model robot
robot = BicycleVehicle(Covariance=V)
rng(0) % obtain repeatable results
% speed is 1 m/s, steered-wheel angle is 0.3
odo = robot.step(1,0.3)
%display the true configuration of robot
robot.q
%assume the previous configuration is [0,0,0]
robot.f([0 0 0],odo)
%create an instance of a driver object and connect it to the robot
robot.addDriver(RandomDriver([-10 10 -10 10],show=true))
robot.run(10);
```

   c) Explanation of calling function:

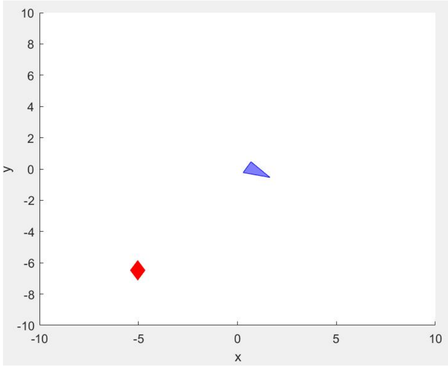
      i. ***robot = BicycleVehicle(Covariance=V)***, which can create a BicycleVehicle object with covariance V.

      ii. ***robot.q*** , which can display its initial state.

      iii. ***odo = robot.step(0.2,0.1)***, which can apply a speed (0.2 m/s) and steered-wheel angle (0.1 rad).

      iv. ***vehicle.addDriver(RandomDriver(10))***, which can add a driver object.

   d) Outcome of experiment:



3. 6.1.2 Estimating Pose

   a) Aim of experiment: Use EKF to estimate robot's pose

   b) Code and comment:

6.1.2 Estimating Pose

```
%compute Fx Jacobians
robot.Fx([0 0 0],[0.5 0.1])
%define the initial covariance
P0 = diag([0.05 0.05 deg2rad(0.5)].^2);
%create a EKF object
ekf = EKF(robot,V,P0);
rng(0) % obtain repeatable results
%Running the filter for 400 time steps
ekf.run(400);
clf
%plot the true path taken by the robot
robot.plotxy
hold on
ekf.plotxy
%the covariance at time step 150
P150 = ekf.history(150).P
sqrt(P150(1,1))
%overlay such ellipses on the plot
ekf.plotellipse(fillcolor="g",alpha=0.3)
clf
ekf.plotP;
```

   c) Explanation of calling function:

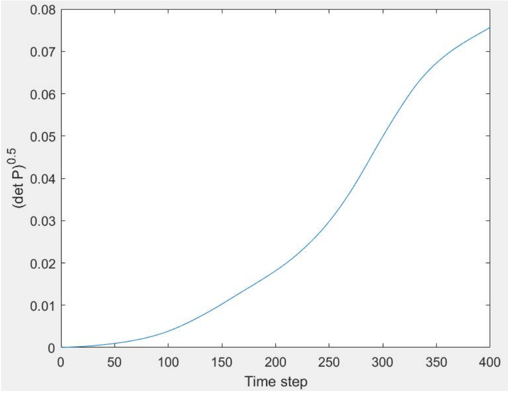
      i. ***J = VEH.Fx(Q, ODO)***, which create Jacobian with the state Q and odometry input ODO.

      ii. ***EKF***, which create EKF object.

      iii. ***robot.plotxy***, which plots true path followed by vehicle.

      iv. ***ekf.plotellipse***, which plot vehicle covariance as an ellipse.

   d) Outcome of experiment:



4. 6.2.1 Localizing in a Landmark Map

   a) Aim of experiment: Use landmark map to localize the robot

   b) Code and comment:

6.2.1 Localizing in a Landmark Map

```
rng(0) % obtain repeatable results
% 20 landmarks and 100 square units
map = LandmarkMap(20,10)
map.plot
%set the sensor uncertainty
W = diag([0.1 deg2rad(1)].^2);
%model this type of sensor
sensor = LandmarkSensor(robot,map,covar=W, ...
  range=10,angle=[-pi/2 pi/2]);
[z,i] = sensor.reading
map.landmark(14)
%%%%%%%%%%%
rng(0) % obtain repeatable results
map = LandmarkMap(20,10);
V = diag([0.02 deg2rad(0.5)].^2);
robot = BicycleVehicle(Covariance=V);
robot.addDriver(RandomDriver(map.dim,show=true));
sensor = LandmarkSensor(robot,map,covar=W, ...
  range=4,angle=[-pi/2 pi/2],animate=true);
P0 = diag([0.05 0.05 deg2rad(0.5)].^2);
ekf = EKF(robot,V,P0,sensor,W,map);
ekf.run(400)
clf
map.plot
robot.plotxy("b");
ekf.plotxy("r");
ekf.plotellipse(fillcolor="g",alpha=0.3)
```

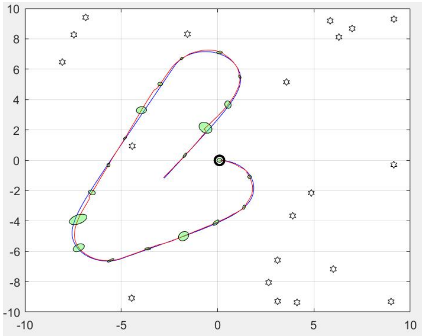
   c) Explanation of calling function:

      i. ***LandmarkMap,*** which is Map of planar point landmarks.

      ii. ***LandmarkSensor,*** which is Range and bearing sensor class.

      iii. ***Landmark***, which return landmarks from map.

   d) Outcome of experiment:



5. 6.2.2 Creating a Landmark Map

   a) Aim of experiment: Use RVC Toolbox to create a landmark map.

   b) Code and comment:

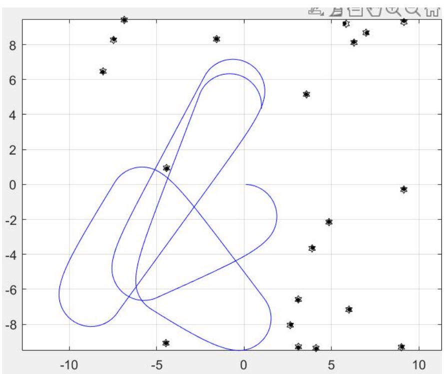
6.2.2 Creating a Landmark Map

```
rng(0) % obtain repeatable results
% set 20 landmarks and 100 square units
map = LandmarkMap(20,10);
% create a class
robot = BicycleVehicle; % error free vehicle
% add a model
robot.addDriver(RandomDriver(map.dim));
% set covariance
W = diag([0.1 deg2rad(1)].^2);
% init sensor class
sensor = LandmarkSensor(robot,map,covar=W);
ekf = EKF(robot,[],[],sensor,W,[]);
% 1000times
ekf.run(1000);
% plot
map.plot;
ekf.plotmap("g");
robot.plotxy("b");
ekf.landmarks(:,17)'  % transpose for display
ekf.x_est(1:2)'  % transpose for display
ekf.P_est(1:2,1:2)
```

   c) Explanation of calling function:

      i. plotmap, which plot self-organizing map.

   d) Outcome of experiment:



6. 6.2.3 EKF SLAM

   a) Aim of experiment:

b) Code and comment:

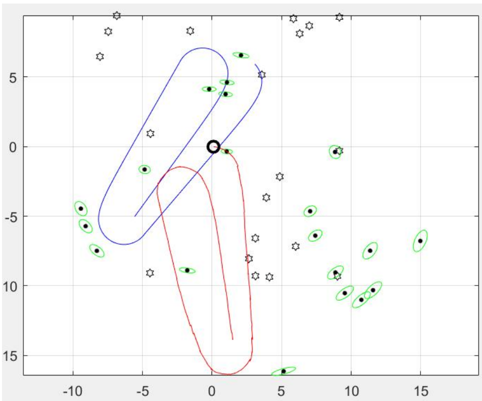
6.2.3 EKF SLAM

```
rng(0) % obtain repeatable results
% set 20 landmarks and 100 square units
map = LandmarkMap(20,10);
% set covariance
V = diag([0.1 deg2rad(1)].^2);
%create BicycleVehicle object to model robot
robot = BicycleVehicle(covar=V, q0=[3 6 deg2rad(-45)]);
%create an instance of a driver object and connect it to the robot
robot.addDriver(RandomDriver(map.dim));
% set covariance
W = diag([0.1 deg2rad(1)].^2);
% init sensor class
sensor = LandmarkSensor(robot,map,covar=W);
% set covariance
P0 = diag([0.05 0.05 deg2rad(0.5)].^2);
% using the default robot configuration
ekf = EKF(robot,V,P0,sensor,W,[]);
ekf.run(500,x_est0=[0 0 0]);
map.plot;            % plot true map
robot.plotxy("b"); % plot true path
ekf.plotmap("g");  % plot estimated landmarks + covariances
ekf.plotxy("r");   % plot estimated robot path
T = ekf.transform(map)
```

c) Explanation of calling function:

i. ***Ekf.transform***, which compute an **SE**(2) transformation from the map frame to the world frame

d) Outcome of experiment:



Experimental Results and Analysis:

1. By Kalman Filter, we get the optimal estimate of temperature indoor. This outcome is close to the simulated real value of temperature indoor.
2. About 6.1.1 Modeling the Robot, we successfully model the motion of robot under the influence of noise, which is placed with covariance.
3. About 6.1.2 Estimating Pose, we successfully use Kalman Filter to estimate the pose of robot while robot is moving.
4. About 6.2.1 Localizing in a Landmark Map, we successfully use new information from estimate of landmark to localize the position of robot.
5. About 6.2.2 Creating a Landmark Map, we finish a landmark map and let robot move on this map from a landmark to next landmark in order.
6. About 6.2.3 EKF SLAM, By EKF SLAM, we tackle the problem of determining our position and creating a map at the same time.

指导教师批阅意见：

成绩评定：

指导教师签字：

年　　月　　日

备注：

注：1、报告内的项目或内容设置，可根据实际情况加以调整和补充。

2、教师批改学生实验报告时间应在学生提交实验报告时间后 10 日内。