**Robotics**

**Exercise 4.1 Composition of poses and landmarks**

The purpose of this exercise is to get familiar with the process of observing landmarks from robot poses. The main tools for that are:

* the **composition of two poses** and the **composition of a pose and a landmark.**
* the **propagation of uncertainty** through the Jacobians of these compositions.

We will address several problems in an incremental complexity way. The following figures will help you to follow the exercise.

*r*

Sensor pose:

Landmark observation:



Landmark observation:

Robot/Sensor pose:



*x*

*y*



1. Let’s consider a robot R1 at a perfectly known pose p1 = [1, 2, 0.5]T which observes a landmark *m* with a range-bearing (polar) sensor affected by a zero-mean Gaussian error with covariance W1p = diag([0.25, 0.04]). The sensor provides the measurement z1p = [4m., 0.7rad.] T. Compute the Gaussian probability distribution (mean and covariance)of the landmark in the world frame (the same as the robot) and plot its corresponding ellipse (in magenta, sigma=1).

*Hint*: Prior to propagate the measurement uncertainty, we need to compute the covariance of the observation in the Cartesian robot R1 frame:



1. Now, let’s assume that the robot pose is not known, but a RV that follows a Gaussian probability distribution: p1 ~ N([1, 2, 0.5]T, 1) with 1 = diag([0.08,0.6, 0.02]).
   1. Compute the covariance matrix m1 of the landmark in the world frame and plot it as an ellipse centered at the mean m1 (in blue, sigma= 1). Plot also the covariance of the robot pose (in blue, sigma= 1).
   2. Compare the covariance with that obtained in the previous case. Is it bigger? Is it bigger than that of the robot? Why?
2. Another robot R2 is at pose p2 ~ ([6m., 4m., 2.1rad.]T, 2) with 2 = diag([0.20,0.09, 0.03]). Plot p2 and its ellipse (covariance) in green (sigma=1). Compute the relative pose p12 between R1 and R2. For that, take a look at the file *“Clarifying the relative pose between to poses”* and implement the two possible ways to obtain such pose.
3. According to the information that we have about the position of the landmark *m* in the world coordinates (provided by R1), compute the predicted observation distribution of *z2p =*[*r*, *α*] ~ N([*z2p*, W2p) by a range-bearing sensor from R2.

*Hint*: We need to compute the covariance of the predicted observation in *Polar* coordinates (W2p). For that, use the following *Jacobian*:

1. Assume now that a measurement z2 = [4m., 0.3rad.] T of the landmark is taken from R2 with a sensor having the same precision as that of R1 (W2p= W1p).
   1. What is the pdf of the observed landmark according to this observation? Plot the corresponding ellipse (in green, sigma=1).
   2. Two different pdf’s are now associated to the same landmark.
      1. Is that a contradiction?
      2. Can you work out a solution that combines these two “pieces of information”? Plot it (in red).

**Results:**

**All the plots of the exercise:**

****

**4.1.1 Sensor measurement in the world's coordinate system (mean and covariance)**

z1\_w = [2.4494, 5.7282, 1.5000]’ (Only the first two values have meaning)

Wz1\_w = [0.5888 -0.1317

-0.1317 0.3012]

**4.1.2 Sensor measurement in the world's coordinate system with uncertainty in the robot pose**

Wz1\_w = [0.9468 -0.2398

-0.2398 0.9432]

**4.1.3 Relative pose (Gaussian distribution) between R1 and R2.**

p12\_w = [5.3468, -0.6420, 1.6000]’

Qp12\_w = [0.3825 0.2411 0.0128

0.2411 1.1675 0.1069

0.0128 0.1069 0.0500]

**4.1.4 Predicted observation of the landmark m by a range-bearing sensor from R2**

Z2p\_r= [3.9488 m., 0.5886rad]

W2\_p = [1.1348 -0.0371

-0.0371 0.0484]

**4.1.5 Sensor measurement from R2 in the world's coordinate system (mean and covariance)**

z2\_w = [3.0504, 6.7019, 3.1000]’ (Only the first two values have meaning)

Wz2c\_w = [0.8469 0.4333

0.4333 0.8131]

**Combined information**

z\_w = [2.5876, 6.1553]’

Wz\_w = [0.3797 0.0777

0.0777 0.3700]

%

% Exercise of the ''Robot sensing'' lecture

% Composition of poses and landmarks

close all

clear var

clc

% Sufix meaning:

% \_w: world reference frame

% \_r: robot reference frame

% Other codes:

% p: in polar coordinates

% c: in cartesian coordinates

% e.g. z1p\_r represents an observation z1 in polar in the robot reference

% frame

%-------------------------------------------------------------------------%

% 4.1.1 %

%-------------------------------------------------------------------------%

% Robot

p1\_w **=** **[**1**,**2**,**0.5**]';** % Robot R1 pose

Qp1\_w **=** zeros**(**3**,**3**);** % Robot pose covariance matrix (uncertainty)

% Landmark

z1p\_r **=** **[**4**,**0.7**]';** % Measurement/observation

W1p\_r **=** diag**([**0.25**,** 0.04**]);** % Sensor noise covariance

% 1. Convert polar coordinates to cartesian (in the robot frame)

z1xc\_r **=** ---------**;**

z1yc\_r **=** ---------**;**

zc\_r **=** **[**z1xc\_r**,** z1yc\_r**];**

% 2. Obtain the sensor/measurement covariance in cartesian coordinates in

% the frame of the robot (it is given in polar). For that you need the

% Jacobian built from the expression that converts from polar to cartesian

% coordinates.

r **=** z1p\_r**(**1**);** % Useful variables

alpha **=** z1p\_r**(**2**);**

c **=** cos**(**alpha**);**

s **=** sin**(**alpha**);**

J\_pc **=** **[**---------**];** % Build the Jacobian

Wzc\_r **=** J\_pc**\***W1p\_r**\***J\_pc**';**

% 3. Ok, we are now ready for computing the sensor measurement in the

% world's coordinate system (mean and covariance).

z1\_w **=** tcomp**(**---------**)** % Compute coordinates of the landmark in the world

J\_ap **=** ---------**;** % Now build the Jacobians

J\_aa **=** ---------**;**

Wzc\_w **=** J\_ap**\***-----**\***J\_ap**'** **+** J\_aa**\***-----**\***J\_aa**'** % Finally, propagate the covariance!

% Draw results

plot**(**z1\_w**(**1**),**z1\_w**(**2**),**'x'**);**

xlim**([-**1**,**10**])**

ylim**([-**1**,**10**])**

grid on**;**

hold on**;**

pbaspect**([**1 1 1**])**

text**(**z1\_w**(**1**)+**1**,**z1\_w**(**2**),**'Landmark'**,**'color'**,**'k'**);**

PlotEllipse**(**z1\_w**(**1**:**2**),**Wzc\_w**,**1**,**'m'**);**

DrawRobot**(**p1\_w**,**'b'**);**

text**(**p1\_w**(**1**)+**1**,**p1\_w**(**2**),**'R1'**,**'color'**,**'b'**);**

%-------------------------------------------------------------------------%

% 4.1.2 %

%-------------------------------------------------------------------------%

% Now, we have uncertainty in the robot pose!

Qp1\_w **=** ---------**;** % New R1 covariance diag(x, y, theta)

% Propagate the covariances again, taken into account this new info.

Wzc\_w **=** ---------------------------

% Draw result

PlotEllipse**(**p1\_w**(**1**:**2**),**Qp1\_w**,**1**,**'b'**);**

PlotEllipse**(**z1\_w**(**1**:**2**),**Wzc\_w**,**1**,**'b'**);**

%-------------------------------------------------------------------------%

% 4.1.3 %

%-------------------------------------------------------------------------%

p2\_w **=** **[**6**,**4**,**2.1**]';** % Pose of the second robot R2

Qp2\_w **=** diag**([**0.20**,**0.09**,**0.03**]);** % Covariance matrix related to the pose

% Draw robot

PlotEllipse**(**p2\_w**(**1**:**2**),**Qp2\_w**,**1**,**'g'**);**

DrawRobot**(**p2\_w**,**'g'**);**

text**(**p2\_w**(**1**)+**1**,**p2\_w**(**2**),**'R2'**,**'color'**,**'g'**);**

% Compute the relative pose between p12 between R1 and R2

% First way: composition of poses with inverse pose

p1inv\_w **=** ---------**;**

p12\_w **=** tcomp**(**---------**)**

Qinvp1\_w **=** Jinv**(**---------**)\***Qp1\_w**\***Jinv**(**---------**)';**

Qp12\_w **=** ------------------------------------**'**

% Second way: Inverse Composition

c **=** cos**(**p1\_w**(**3**));** % Useful variables

s **=** sin**(**p1\_w**(**3**));**

xp1 **=** p1\_w**(**1**);** yp1 **=** p1\_w**(**2**);**

xp2 **=** p2\_w**(**1**);** yp2 **=** p2\_w**(**2**);**

J\_p12p1 **=** **[**---------**];**

J\_p12p2 **=** **[**---------**];**

Qp12\_w **=** ------------------------------------

%-------------------------------------------------------------------------%

% 4.1.4 %

%-------------------------------------------------------------------------%

% 1. Take a measurement using the range-bearing observation model!

r2 **=** ---------**;**

alpha2 **=** ---------**;**

z2p\_r **=** **[**r2**,**alpha2**]'**

% 2. Jacobian from cartesian to polar at z2p\_r when the covariance is in

% global coordianes

alpha **=** alpha2 **+** p2\_w**(**3**);**

Jcp\_p2 **=** **[**---------**];**

% 3. Finally, propagate the uncertainty to polar coordinates in the

% robot frame

W2\_p **=** --------- %dim: 2x2

%-------------------------------------------------------------------------%

% 4.1.5 %

%-------------------------------------------------------------------------%

z2p\_r **=** **[**4**,**0.3**]';**

W2p\_r **=** diag**([**0.25**,** 0.04**]);** % Sensor noise covariance

% 1. Convert polar coordinates to cartesian (in the robot frame)

x2 **=** ---------**;**

y2 **=** ---------**;**

z2c\_r **=** **[**x2**,**y2**];**

% 2. Obtain the sensor/measurement covariance in cartesian coordinates in

% the frame of the robot (it is given in polar). For that you need the

% Jacobian built from the expression that converts from polar to cartesian

% coordinates.

r **=** z2p\_r**(**1**);**

alpha **=** z2p\_r**(**2**);**

c **=** cos**(**alpha**);**

s **=** sin**(**alpha**);**

J\_pc **=** **[**---------**];**

Wz2c\_r **=** J\_pc**\***W2p\_r**\***J\_pc**';**

% 3. Ok, we are now ready for computing the sensor measurement in the

% world's coordinate system (mean and covariance).

J\_ap **=** ---------**;**

J\_aa **=** ---------**;**

Wz2c\_w **=** J\_ap**\***---------**\***J\_ap**'** **+** J\_aa**\***---------**\***J\_aa**'**

z2\_w **=** tcomp**(**---------**)** % Compute coordinates of the landmark in the world

% Draw result

plot**(**z2\_w**(**1**),**z2\_w**(**2**),**'xg'**);**

PlotEllipse**(**z2\_w**(**1**:**2**),**Wz2c\_w**,**1**,**'g'**);**

% 4. Combine the measurements from both sensors!

Wz\_w **=** ---------

z\_w **=** ---------

% Draw result

plot**(**z\_w**(**1**),**z\_w**(**2**),**'xr'**);**

PlotEllipse**(**z\_w**(**1**:**2**),**Wz\_w**,**1**,**'r'**);**

%-------------------------------------------------------------------------%