Kalman Filter for Mobile Robot Localization

Who? Paulo Pinheiro (T.A) / Eleri Cardozo (Professor)

From? 2014 Class of IA368W: Métodos Estocásticos em Robótica Móvel (UNICAMP)

pinheiro@ic.unicamp.br

When? May 15, 2014

Instructions

- THIS GUIDE IS INTENDED TO HELP THE IA368W CLASS DEVELOP THE KALMAN FILTER.
- The realm is mobile robot localization problem.
- The equations are colorful for an easy understanding.
- We assume you have the robot with a laser and odometry information.
  - The odometry must return [x,y,th].
  - The laser must return a set of detected features.
    - For each laser reading you should get from your feature detection algorithm the [Lx, Ly, lx, ly], where Lx, Ly is the real pose of the landmark, and lx,ly, the pose of the correspondent landmark the robot estimated.
  - The algorithm should work fine even with a single feature.
- For each element in the equations, we will show an example of expected output, thus you can be aware of what kinda matrix or value you should expect for.
- If you find some issue/typo in this presentation or want to make some suggestion, let me know. pinheiro@ic.unicamp.br

Algorithm - This is all you need to get it done!

while *true* **do**

```
// Reading robot's pose
```

PoseR = GET[Pose.x; Pose.y; Pose.th]

```
// Prediction step
```

$\bar{\Sigma}_t = (G_t * \Sigma_{t-1} * G_t^T) + (V_t * \Sigma_{\Delta_t} * V_t^T) + Rt$

```
// Update step
```

feature = FeatureDetection;

$K_t = \bar{\Sigma}_t * H_t^T * (H_t * \bar{\Sigma}_t * H_t^T + Q_t)^{-1}$

$INOVA = (zsensor_t - zreal_t)$

$X_t = \bar{X}_t + K_t * INOVA$

$PoseR = X_t$ and $PUT(PoseR)$

$\Sigma_t = (I - K_t H_t)\bar{\Sigma}_t$

$sleep(Dt)$

end

Equation 1 - Getting the pose of the robot

$$\text{PoseR} = \text{GET}[\text{Pose.x; Pose.y; Pose.th}]$$

Equation 1 - Getting the pose of the robot

PoseR = GET[Pose.x; Pose.y; Pose.th]

- The odomotry must return the pose of the robot. It's ok if it is not so precise.

Matlab example

```
Pose = GET([host '/motion/pose']);
Pose.th = mod(Pose.th*pi/180,2*pi);
PoseR = [Pose.x; Pose.y; Pose.th];
```

Example what you should get:

$$PoseR = (2373, 1608, 6)$$

Equation 2 -Starting prediction stage.

$$\bar{\Sigma}_t = (G_t * \Sigma_{t-1} * G_t^T) + (V_t * \Sigma_{\Delta_t} * V_t^T) + Rt$$

- $\bar{\Sigma}_t$ is the uncertainty over the pose.

Equation 2

$$\bar{\Sigma_t} = (G_t * \Sigma_{t-1} * G_t^T) + (V_t * \Sigma_{\Delta_t} * V_t^T) + Rt$$

G_t

$$G_t = \begin{bmatrix} 1 & 0 & -\Delta s_t \sin(\theta_{t-1} + \frac{\Delta\theta_t}{2}) \\ 0 & 1 & \Delta s_t \cos(\theta_{t-1} + \frac{\Delta\theta_t}{2}) \\ 0 & 0 & 1 \end{bmatrix}$$

V_t

$$V_t = \begin{bmatrix} a & c \\ b & d \\ \frac{1}{2b} & -\frac{1}{2b} \end{bmatrix}$$

$$a = \frac{1}{2}\cos(\theta_{t-1} + \frac{\Delta\theta_t}{2}) - \frac{\Delta st}{4b}\sin(\theta_{t-1} + \frac{\Delta\theta_t}{2})$$
$$b = \frac{1}{2}\sin(\theta_{t-1} + \frac{\Delta\theta_t}{2}) + \frac{\Delta st}{4b}\cos(\theta_{t-1} + \frac{\Delta\theta_t}{2})$$
$$c = \frac{1}{2}\cos(\theta_{t-1} + \frac{\Delta\theta_t}{2}) + \frac{\Delta st}{4b}\sin(\theta_{t-1} + \frac{\Delta\theta_t}{2})$$
$$d = \frac{1}{2}\sin(\theta_{t-1} + \frac{\Delta\theta_t}{2}) - \frac{\Delta st}{4b}\cos(\theta_{t-1} + \frac{\Delta\theta_t}{2})$$

New Variables...

θ_{t-1}? $\Delta\theta_t$? Δs_t? and b? - Next Slide.

A stop by to define the new variables...

θ_{t-1}	Robot's current theta. PoseR(3).
$\Delta\theta_t$	$\Delta\theta_t = (Vel.right * Dt - Vel.left * Dt)/(2 * b)$.
Δs_t	$\Delta s_t = (Vel.right * Dt + Vel.left * Dt)/2$.
b	The axis of robot. Let's use $b = 165$.
$Vel.right$ and $Vel.left$	Velocity of right wheel and left wheel.
Dt	Time the loop will sleep. Let's use 2 seconds. $Dt = 2$.

Keep going on Equation 2

$$\bar{\Sigma_t} = (G_t * \Sigma_{t-1} * G_t^T) + (V_t * \Sigma_{\Delta_t} * V_t^T) + Rt$$

- The initial pose covariance Σ_{t-1} or Σ_0 is a 3x3 matrix of zeros.
- It will be updated by the last equation of the algorithm.

Σ_{t-1}

$$\Sigma_{t-1} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

Σ_{Δ_t}

$$\Sigma_{\Delta_t} = \begin{bmatrix} K_s|\Delta s_t| & 0 \\ 0 & K_\theta|\Delta\theta_t| \end{bmatrix}$$

K_s $K_s = 0.1$

K_θ $K_\theta = 0.1$

- Both K_s and K_θ are good values for the robot used in the classes.

Equation 2

$$\bar{\Sigma_t} = (G_t * \Sigma_{t-1} * G_t^T) + (V_t * \Sigma_{\Delta_t} * V_t^T) + Rt$$

R_t

$$R_t = \begin{bmatrix} \sigma_x^2 & 0 & 0 \\ 0 & \sigma_y^2 & 0 \\ 0 & 0 & \sigma_\theta^2 \end{bmatrix}$$

σ_x Laser error for x: $\sigma_x = 5$

σ_y Laser error for y: $\sigma_y = 5$

σ_θ Laser error for theta: $\sigma_\theta = 1$

- All σ are good values for the robot used in our classes.

Equation 2

$$\bar{\Sigma_t} = (G_t * \Sigma_{t-1} * G_t^T) + (V_t * \Sigma_{\Delta_t} * V_t^T) + Rt$$

You should get something like that:

Example G:

$$G = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Example Σ_{t-1}:

$$\Sigma_{t-1} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

Example V_t:

$$V = \begin{bmatrix} 0.4999194 & 0.4999194 \\ -0.0089757 & -0.0089757 \\ 0.0030303 & -0.0030303 \end{bmatrix}$$

Example Σ_{Δ_t} (robot is not moving):

$$\Sigma_{\Delta_t} = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}$$

Equation 2

$$\bar{\Sigma_t} = (G_t * \Sigma_{t-1} * G_t^T) + (V_t * \Sigma_{\Delta_t} * V_t^T) + Rt$$

You should get something like that (continuation):

Example Rt:

$$Rt = \begin{bmatrix} 25 & 0 & 0 \\ 0 & 25 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Example $\bar{\Sigma_t}$

$$\bar{\Sigma_t} = \begin{bmatrix} 25 & 0 & 0 \\ 0 & 25 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

- Since is the 1st run, the $\bar{\Sigma_t}$ has the value of R_t

Equation 3 - Feature Detection.

$$feature = FeatureDetection;$$

Equation 3 - Feature Detection.

feature = FeatureDetection;

- The FeatureDetection function should return the landmark pose captured by the laser (l_x, l_y) and the correspondent real landmark pose (L_x, L_y).

Example of what you could get:

$$feature = (l_x, l_y, L_x, L_y)$$
$$feature = (4704, 665, 4700, 670)$$

Equation 4 - Calculating the factor K_t (Update stage)

$$K_t = \bar{\Sigma}_t * H_t^T * (H_t * \bar{\Sigma}_t * H_t^T + Q_t)^{-1}$$

Be prepared! The size of the matrices H and Q will depend on the number of features you've got. Let's take a look...

Equation 4 - Calculating the factor K_t.

$$K_t = \bar{\Sigma}_t * H_t^T * (H_t * \bar{\Sigma}_t * H_t^T + Q_t)^{-1}$$

H_t For each feature, it will be required these 2 rows in H.

$$H_t = \begin{bmatrix} -\frac{L_x - x_t}{\sqrt{q}} & -\frac{L_y - y_t}{\sqrt{q}} & 0 \\ \frac{L_y - y_t}{q} & -\frac{L_x - x_t}{q} & -1 \end{bmatrix}$$

q $q = (L_x - x_t)^2 + (L_y - y_t)^2$

x_t and y_t x_t is the x of the robot pose (PoseR(1))
y_t is the y of the robot pose (PoseR(2))

Equation 4 - Calculating the factor K_t

$$K_t = \bar{\Sigma}_t * H_t^T * (H_t * \bar{\Sigma}_t * H_t^T + Q_t)^{-1}$$

Q_t For each feature, it will be required these 2 rows in Q (example for 1 feature).

$$Q_t = \begin{bmatrix} \sigma_{ld}^2 & 0 \\ 0 & \sigma_{l\theta}^2 \end{bmatrix}$$

Q_t Example for 2 features.

$$Q_t = \begin{bmatrix} \sigma_{ld}^2 & 0 & 0 & 0 \\ 0 & \sigma_{l\theta}^2 & 0 & 0 \\ 0 & 0 & \sigma_{ld}^2 & 0 \\ 0 & 0 & 0 & \sigma_{l\theta}^2 \end{bmatrix}$$

σ_{ld} $\sigma_{ld} = 0.5$ (laser distance accuracy related)

$\sigma_{l\theta}$ $\sigma_{l\theta} = 0.1$ (laser angular accuracy related)

- Both $\sigma_{l\theta}$ and σ_{ld} are good values for our robot.

$\bar{\Sigma}_t$ $\bar{\Sigma}_t$ We already got it from the equation 2.

Equation 4 - Calculating the factor K_t

$$K_t = \bar{\Sigma}_t * H_t^T * (H_t * \bar{\Sigma}_t * H_t^T + Q_t)^{-1}$$

For 1 feature, you should get something like that:

Example H_t:

$$H_t = \begin{bmatrix} -0.82887 & -0.55944 & 0.00000 \\ 0.00025 & -0.00037 & -1.00000 \end{bmatrix}$$

Example Q_t:

$$Q_t = \begin{bmatrix} 0.25 & 0 \\ 0 & 0.01 \end{bmatrix}$$

Example K_t for 1 feature.

$$K_t = \begin{bmatrix} 0.71 & -0.007 \\ 0.68 & -0.007 \\ 0 & -0.99 \end{bmatrix}$$

Example K_t for 2 features.

$$K_t = \begin{bmatrix} -0.55 & -0.004 & -0.01 & 0.012 \\ -0.8 & -0.001 & 0.23 & 0.001 \\ 0.003 & -0.34 & -0.00004 & -0.033 \end{bmatrix}$$

Equation 5 - Calculating the Innovation value

$$INOVA = [zsensor_t - zreal_t]$$

Equation 5 - Calculating the Innovation value

$$INOVA = [zsensor_t - zreal_t]$$

$zsensor_t$ — $zsensor_t$ is related to the information the feature detection gave to you.

$$zsensor_t = \begin{bmatrix} zrange \\ zbearing \end{bmatrix}$$

$zrange$ — $\sqrt{(l_x - PoseR(1))^2 + (ly - PoseR(2))^2)}$

$zbearing$ — $arct2((ly - PoseR(2)), (l_x - PoseR(1))) - PoseR(3);$

l_x and l_y — l_x and l_y are the values returned from the FeatureDetection. They are related to the point where the laser thinks the landmark is at. Not the real pose of the landmark.

Equation 5 - Calculating the Innovation value

$$INOVA = [zsensor_t - zreal_t]$$

$zreal_t$ — $zreal_t$ is related to the real Landmark correspondent to the z_t

$$zreal_t = \begin{bmatrix} Lrange \\ Lbearing \end{bmatrix}$$

$Lrange$ — $\sqrt{(L_x - PoseR(1))^2 + (Ly - PoseR(2))^2)}$

$Lbearing$ — $arct2((Ly - PoseR(2)), (L_x - PoseR(1))) - PoseR(3);$

L_x and L_y — L_x and L_y are the values returned from the FeatureDetection. They compose the real pose of the correspondent real Landmark.

Equation 5 - Calculating the Innovation value

$$INOVA = [zsensor_t - zreal_t]$$

You should get something like that:

- For each feature a 2x1 matrix is added.

Example *INOVA* for 1 feature:

$$INOVA = \begin{bmatrix} 3.00 \\ 0.000003 \end{bmatrix}$$

Example $K_t * INOVA$ for 1 feature

$K_t * INOVA = (-2, 2, 0)$
This is the pose difference you must add to the robot's pose.

Equation 6 - Updating the new pose

$$PoseR = X_t + K_t * INOVA \text{ and } PUT(PoseR)$$

Equation 6 - Updating the new pose

$PoseR = X_t + K_t * INOVA$

$PoseR = X_t + K_t * INOVA$ and $PUT(PoseR)$

$PoseR = PoseR + K_t * INOVA;$
$NewPose.x = PoseR.x;$
$NewPose.y = PoseR.y;$
$NewPose.th = PoseR.th * 180/pi;$

PUT([host '/motion/pose'], NewPose);

Equation 7 - Updating the pose covariance Σ_t

$$\Sigma_t = (I - K_t H_t)\bar{\Sigma}_t$$

Equation 7 - Updating the pose covariance Σ_t

$$\Sigma_t = (I - K_t H_t)\bar{\Sigma}_t$$

I

$$I = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

K_t We've already got it. See the Equation 4.

H_t We've already got it. See the Equation 4.

$\bar{\Sigma}_t$ We've already got it. See the Equation 2.

Example Σ_t:

$$\Sigma_t = \begin{bmatrix} 1.3038e-01 & 5.2667e-02 & -2.7058e-05 \\ 5.2667e-02 & 2.8502e-01 & -1.0314e-04 \\ -2.7058e-05 & -1.0314e-04 & 3.3223e-03 \end{bmatrix}$$

Equation 8 - Sleeping for the next round

$$sleep(Dt)$$

Equation 8 - Sleeping for the next round

$sleep(Dt)$

- Sleep for n seconds. Let's use $Dt = 2$.
- Done.

Then Then do it all again.

END.