Raspberry Pi GoPiGo Robot EKF SLAM Implementation

Gopal Menon
Computer Science Department
Utah State University
Logan, Utah 84322
Email: gopal.menon@aggiemail.usu.edu

***Abstract*—Robot motion is stochastic and not deterministic. A robot does not faithfully execute movement commands due to inaccuracies in its motion mechanism, approximations made in modeling its environment and incomplete modeling of the physics of its operating environment. Due to these reasons, the robot may not end up exactly at the location where it should have been if the movement command had been faithfully executed. The robot position may be thought of as a two dimensional probability distribution. This probability distribution will have a mean that will be equal to the expected robot location based on the movement command. The covariance will be the measure of how faithfully it executes commands. Due to this uncertainty in movement, with each motion command, the location uncertainty increases and after a while the robot gets lost. In some applications, robots need to be able to go into an unknown environment and explore it. This is a difficult problem due to the issue described above. If a map of the environment is available, the robot can use sensors to find its location in the environment. Conversely, if the robot knows where it is, it can generate an environment map. This is a sort of chicken and end problem. Simultaneous Localization and Mapping (SLAM) solves this problem. An Extended Kalman Filter (EKF) can be used to reduce robot location uncertainty. This document describes the implementation of EKF SLAM using a robot.**

I. Introduction

A. The robot platform

The robot used for this implementation is based on a Raspberry Pi and is manufactured and sold by Dexter Industries [1]. The robot is equipped with an ultrasonic sensor that is capable of measuring distances accurately to 250 cm. The sensor is mounted on a servo mechanism that can point the sensor in any direction ahead of the robot. The robot moves through use of two wheels that can be independently controlled by motors. Each wheel has an encoder using which the robot can issue commands for a wheel to move a fixed number of steps. The robot also has a third castor wheel that can move in any direction, but is not powered. The robot will be given commands to make it move and it can sense its surroundings using the ultrasonic sensor.

B. EKF SLAM

A Kalman filter is a gaussian filter algorithm that can use two gaussian distributions for a computed value and merge them together to form a distribution that has lesser variance than either of the two distributions. A Kalman filter however only works in the case of systems where the mechanics are linear. This is where the Extended Kalman Filter (EKF) comes in. An EKF uses an approximation to find the resultant distribution after a prior dostribution is transformed by a non-linear function.

The robot will compute its expected position based on its position and the movement command. It will also make independent measurements of its position using sensors. These two measurements will be combined through use of an EKF in such a way that the position uncertainty is reduced. The Kalman Gain or the reduction in the robot position uncertainty is inversely proportional to the sensor uncertainty. The better the sensor, the higher will be the Kalman Gain and the less uncertainty there will be in the robot position[5].

II. Problem description

A. The Bayes Filter

The Bayes Filter is an algorithm used for calculating beliefsthrun. In our case it is the belief of the robot position. This algorithm is used to compute the belief probability distribution bel from measurement and control data. The measurement data is obtained from sensors and the control data is obtained from the robot command or telemetry.

Bayes Rule says that

$$p(x|y) = \frac{p(y|x)p(x)}{p(y)} \tag{1}$$

Since the denominator in Bayes Rule does not depend on x, the factor $\frac{1}{p(y)}$ can be written as η. So Bayes Rule in 1 can be written as

$$p(x|y) = \eta p(y|x)p(x) \tag{2}$$

Bayes Rule 1 can also be written as

$$p(x|y,z) = \frac{p(y|x,z)p(x|z)}{p(y|z)} \tag{3}$$

as long as $p(y|z) > 0$.

Applying 3 on the robot position x_t gives us

$$p(x_t|z_{1:t}, u_{1:t}) = \frac{p(z_t|x_t, z_{1:t-1}, u_{1:t})p(x_t|z_{1:t-1}, u_{1:t})}{p(z_t|z_{1:t-1}, u_{1:t})} = \eta p(z_t|x_t, z_{1:t-1}, u_{1:t})p(x_t|z_{1:t-1}, u_{1:t}) \tag{4}$$

Since the measurement z_t depends on position x_t and any past measurement $z_{1:t}$ or command $u_{1:t}$ does not provide any additional information on the measurement, we can say that

$$p(z_t|x_t, z_{1:t-1}, u_{1:t}) = p(z_t|x_t) \tag{5}$$

This allows us to simplify 4 as

$$p(x_t|z_{1:t}, u_{1:t}) = \eta p(z_t|x_t)p(x_t|z_{1:t-1}, u_{1:t}) = \eta p(z_t|x_t)\overline{bel}(x_t) \tag{6}$$

where

$$\overline{bel}(x_t) = p(x_t|z_{1:t-1}, u_{1:t}) \tag{7}$$

Using the theorem of total probability

$$p(x) = \int p(x|y)p(y)dy \tag{8}$$

equation 7 can be written as

$$\overline{bel}(x_t) = \int p(x_t|x_{t-1}, z_{1:t-1}, u_{1:t})p(x_{t-1}|z_{1:t-1}, u_{1:t})dx_{t-1} \tag{9}$$

If we know the state x_{t-1}, past measurements and controls have no bearing on the state x_t. So

$$p(x_t|x_{t-1}, z_{1:t-1}, u_{1:t}) = p(x_t|x_{t-1}, u_t) \tag{10}$$

We can see that command u_t can be omitted from the set of conditioning variables in $p(x_{t-1}|z_{1:t-1}, u_{1:t})$. This gives us the recursive update equation 7 as

$$\overline{bel}(x_t) = \int p(x_t|x_{t-1}, u_t)p(x_{t-1}|z_{1:t-1}, u_{1:t-1})dx_{t-1} \tag{11}$$

This reduces to

$$\overline{bel}(x_t) = \int p(x_t|x_{t-1}, u_t)bel(x_{t-1})dx_{t-1} \tag{12}$$

B. The Bayes Filter Algorithm

Based on the above mathematical derivation for a Bayes Filter, we can write down the Bayes Filter Algorithm as follows

Algorithm 1 Bayes Filter Algorithm

1: **procedure** BAYESFILTERALGORITHM($bel(x_{t-1}, u_t, z_t)$)
2:     **for** all x_t **do**
3:         $\overline{bel}(x_t) = \int p(x_t|x_{t-1}, u_t)bel(x_{t-1})dx_{t-1}$
4:         $bel(x_t) = \eta p(z_t|x_t)\overline{bel}(x_t)$
5:     **end for**
6: **end procedure**

The belief $bel(x_t)$ at time t is computed from the belief $bel(x_{t-1})$ at time $t-1$. The input to the algorithm is the belief bel at time $t-1$, along with the most recent control value u_t and the most recent measurement z_t. Its output is the belief $bel(x_t)$ at time t.

The Bayes Filter Algorithm consist of two essentials steps. In line 3, which is the prediction step, it calculates its belief over state x_t based on the prior belief over state x_{t-1} and the control u_t. The belief $\overline{bel}(x_t)$ that the robot assigns to state x_t is obtained by the integral (or in other words the sum) of the product of two distributions: the prior assigned to x_{t-1}, and the probability that control u_t induces a transition from x_{t-1} to x_t.

The second step of the Bayes Filter is called the measurement update. In line 4, the Bayes Filter algorithm multiplies the belief $\overline{bel}(x_t)$ by the probability that the measurement z_t may have been observed. It does this for every hypothetical posterior state x_t.

C. The Kalman Filter

D. The Extended Kalman Filter

The robot state transition function and landmark measurement probabilities are governed by nonlinear functions g and h respectively[4]:

$$x_t = g(u_t, x_{t-1}) + \epsilon_t \tag{13}$$

$$z_t = h(x_t) + \delta_t \tag{14}$$

Here x_t is the robot location at time t, u_t is the robot command that causes it to reach position x_t, z_t is the landmark measurement at time t, ϵ_t is the robot motion uncertainty that depends of the mechanical properties of the robot and δ_t is the sensor measurement uncertainty.

III. Approach/Methods

IV. Results

V. Discussion

VI. Future work

References

[1] GoPiGo - Dexter Industries. *Dexter Industries*. N.p., n.d. Web. 06 Nov. 2015. http://www.dexterindustries.com/gopigo/.

[2] Durrant-Whyte, H., and T. Bailey, *Simultaneous Localization and Mapping: Part I*, IEEE Robotics and Automation Magazine. 13.2 (2006): 99-110.

[3] Bailey, Tim, and Hugh Durrant-Whyte, *Simultaneous localization and mapping (SLAM): Part II*, IEEE Robotics and Automation Magazine. 13.3 (2006): 108-117.

[4] Thrun, Sebastian, Wolfram Burgard, and Dieter Fox, *Probabilistic Robotics*, MIT press, 2005.

[5] Stachniss, Cyrill, *SLAM Course - WS13/14*, YouTube. YouTube, n.d. Web. 18 Sept. 2015. https://www.youtube.com/playlist?list=PLgnQpQtFTOGQrZ4O5QzbIHgl3b1JHimN_.