Simplified occupancy grid mapping

system Nov 2014 post #1

I have a setup, where a distance sensor, such as sonar or IR, rotates about 180 degrees and takes a distance measurement every few degrees. What I would like to do is to create a occupancy grid out of these measurements on an Arduino or, more specifically, I would like to compare two different algorithms, which create the grids.

One of these algorithms would be made by myself, but in order to make a solid comparison, I would like the other algorithm to be the one which is mentioned in occupancy grid materials and research papers, which utilizes a binary Bayes filter. Information about the algorithm and the implementation is available on **Wikipedia** and **here** (slides 9-13).

I'm not too familiar with the technical and the mechanical concepts involved in the binary Bayes filter and its implementation in the algorithm, so bear with me: The main problem for me is that the algorithm involving the binary Bayes filter takes into account "the robot poses", which, as I understand, pertains to the route which the robot has taken. However, my setup with the swiveling sensor is static - the sensor just rotates, but it doesn't change its position in space.

My question is this: if a have a "static" sensor, can I use the binary Bayes filter and the associated algorithm in order to create an occupancy grid? Would this require a small adjustment in the algorithm/math in this case? Would this be feasible on an Arduino? And does using this algorithm in a setup like this even make sense?

Perhaps, I have misunderstood the meaning of the robot poses; either way, any help would be massively appreciated!

system Nov 2014 post #2

From what I have briefly read, it appears to me that the entire purpose of the Bayes filter and occupancy grid mapping is to account for the changing position of the sensor doing the measurement. If the sensor doing the measurement is not moving, then the filtering will still work but its overkill because its not able to perform its function.

Assuming a stationary sensor, you can just take the polar coordinates that you get from your servo's angle and the distance to target and convert it to Cartesian coordinates with some trig. Then you have a grid of objects in the space. Filtering isn't needed for this setup because the only variation will be the resolution/accuracy of your servo and distance measurement. Finding the mean of a sample set can fix this, and some calibration could null out any other issues.

I hope this is helpful.

system Nov 2014 post #3

Thanks for the answer!

I realize that filtering isn't necessary for a setup like this, however, I would like to compare the 'more simple' and the filtering algorithm to see if there's any improvement when using the more 'professional' one.

When you said that it's not going to be able to perform its function, are you saying that the algorithm involving filtering won't work or that it's not going to be utilized in its full potential?

system Nov 2014 post #4

I meant that it will not be utilized to its full potential. The analogy would be like putting wheels on something you never plan to move. It works, but its not necessary.

In terms of implementation, I would look at the arduino code for some of the Kalman filters built for 9-Axis IMUs:

> **MPU-9150_Breakout/firmware at master · sparkfun/MPU-9150_Breakout**
>
> **master/firmware**
>
> Example code and PCB design files for the MPU-9105, 9DOF.

Although used for different things, the Kalman filter appears to function in a similar way to the Bayes filter, so the code may help show you how to accomplish the math functions in arduino that you need.

system Nov 2014 post #5

It's not going to be utilized to its full potential, but it's going to be utilized at least to some extent, unlike like the wheels in your analogy, right?

Thanks for the resource, I'll peruse it later when I'll be starting to actually implement the filter/algorithm.

Another question: would in a setup like this, where there is a single degree of freedom (the angle of the servo), using a Kalman filter make more sense/any at all when compared to a binary Bayes filter? Or does the existence of 1 DOF make them both equally over-kill?

system Nov 2014 post #6

Maybe a poor analogy on my part, I was implying that the wheels would serve to support something like feet would, but they would never be used as wheels. Just as feet, so not fully utilized.

Both are overkill for this setup. A Kalman relies on multiple measurement methods with different error distributions, so you would need multiple sensors. So in a 9-Axis IMU, the accelerometer, magnetometer, and gyros have different error characteristics. Namely, gyros have slow drift over time, and accerlerometers have high noise but over the long run average to roughly the correct value. Combining these two different errors in a Kalman filter provides a real time value with error compensation.

Closed on May 5, 2021

Related topics

Topic	Replies	Views	Activity
Occupancy grid mapping	1	1.4k	Nov 2014

Skip to main content

Topic	Replies	Views	Activity
Particle Filter with Sensor Inputs	11	4.8k	Sep 2017
grid based navigation using ultrasonic sensor	23	13.3k	Dec 2010
Approach for detecting movement, identifying source of movement and aiming servo	10	3.2k	Apr 2015
9250 Yaw measurement filtering	6	364	Nov 2023

Back to top