Red Orange

Magnetic Navigation

By Bennett Brain, ECE '20

Introduction

The goal of magnetic navigation is to use magnetic anomalies to determine the location of aircraft while flying, similar to GPS. However, where GPS relies on satellites, magnetic navigation relies only on its own algorithms and knowing the local magnetic anomalies (that information is available from government research and can be freely downloaded and stored). In simple terms, the idea with magnetic navigation is to start with an inertial measurement unit (a.k.a. an IMU- something virtually all vehicles are equipped with) to estimate position from velocity and acceleration, and then correct those estimates by seeing how well they match up to the magnetic readings from a magnetometer. The anomalies are due to deposits of magnetic materials (such as iron) beneath the Earth's surface, which can be detected at flight heights and are strong enough to be very robust to any intentional interference- to block the effect of a single magnetic deposit one would need to place the equivalent of a power plant over it.

A lot of work needs to go into ensuring that the readings from the magnetometer are accurate and only reflect the local magnetic anomalies, not sources of interference such as power lines. However, the focus of this paper is on the Particle Filter, which is a tool that is used to take post-processed sensor readings and use them to estimate location of an object. This is a key step in magnetic navigation, as it is the workhorse that combines the IMU data and the magnetic data into an accurate estimate of position.

Particle Filtering- *Basics*

The particle filter can be explained as weighting observations of the system based on a model of the system and knowledge of the previous state, and continuously iterating on those estimations, as shown in the following picture(4):

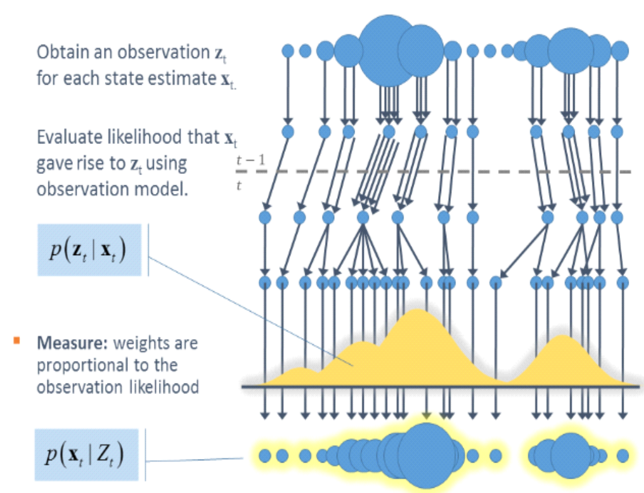

Figure 1. Iterative step in particle filter

Car Analogy

To help understand this, let's walk through how the particle filter works with estimating the position of a car, knowing only a rough estimate of velocity and a more accurate variable that results from position, such as the amount of light directly in front of the car at night (this will increase as you near a streetlight and decrease as you go away from one, but doesn't tell you much on its own because there are many indistinguishable streetlights). First, the system knows the starting position of the car- i.e. the driveway of the McDonald's on the Alewife Parkway. Then, the IMU in the car has a rough estimate of the car's velocity. When the car starts to move, it will predict the location of the car in the next period of time- let's say 5 seconds (in reality, each iteration will be far faster). Knowing that the car is starting to head out of the driveway at 5 feet per second, it predicts the car will be 5 feet away from its starting position at the next time measurements are taken. However, the programmers of this car knew that velocity doesn't give the full picture, so they added in a measure of uncertainty- thus the system will make a lot of predictions that average roughly 5 feet away, but vary around that position. Each of those predictions will be stored and will constitute a "particle".

When that time occurs, the car calculates what the light intensity should be if it were actually 5 feet away (or however far each particle is) from the starting position, as well as measures what the light intensity actually is. It will compare each particle's estimate to the actual value, and whichever particles are closest will get a better "weighting". Note that this does require knowledge of where the street lights are, however road plans are available and these lamp locations can be stored in the car's memory beforehand.

When the car needs to make its next set of predictions, it will take particles with a higher weighting and make its next predictions- the second round of particles, based more on those heavily-weighted round one particles. Lower-weighted particles may still be used as a base for some predictions, but not as many, and the lowest-weighted particles will likely get no predictions made from them at all. At the 10 second mark, this step will repeat- the second set of particles will be again weighted, and then used to generate the third round, and so on. At any given time, the weighted average of the particles can be used to estimate the true position of the car.

Application in Magnetic Navigation

In magnetic navigation, the idea is much the same. However, instead of distance down a highway, the desired measurement is location of an aircraft. And the IMU will measure velocity, acceleration, and many other variables relevant to aircraft (yaw, pitch, roll, etc.). And, most importantly, instead of using street light intensity, the aircraft will use the strength of the local magnetic anomalies. Save for these substitutions, the particle filter works much the same way.

Example results

This implementation can be seen in the particle filter implementation produced by Marin[3]. The results of using a simple simulated example can be seen below:

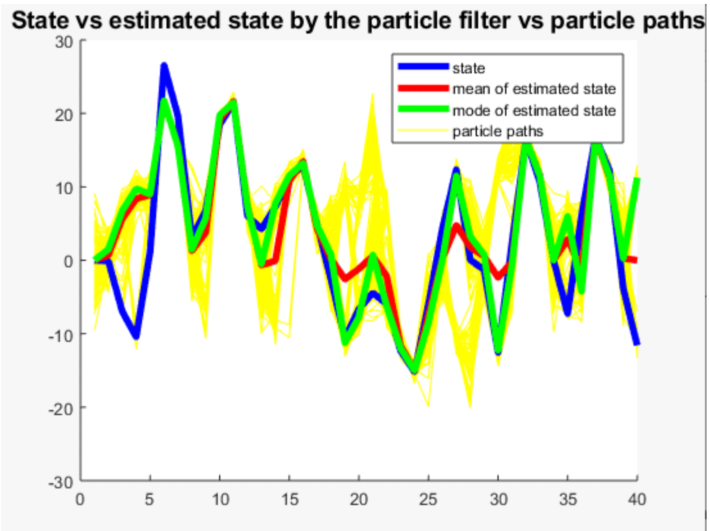

Figure 2. Simulated particle filter results

From Fig. 2, it can be seen that the particles follow the true state, and thus this implementation of the particle filter works. Importantly, after straying from the true path (such as around the start, or near the 20 second mark), the particles are able to return in relatively short order. This shows that the filter is unlikely to have compounding error. The mode is simply the most common state estimate, while the mean is the average state estimate. The mode appears to follow the true state more closely, as outliers don't affect it, but when a lot of particles are off-path, the mean is a better estimator since the on-path particles weight it while they don't affect the mode.

Conclusion

In summary, the particle filter is a key step in implementing magnetic navigation, as it does the final major step of bringing all the measurements and signal processing together into a final determination of the position of the aircraft. However, while many of these steps may be easy to explain, implementation will be challenging. For example, generating the equations needed to estimate the readings for any specific point during flight is a notable hurdle. However, it is clear that once the prerequisite challenges are dealt with, the particle filter will be incredibly useful.

References

[1] [Babb, 2015] Babb, T. (2015). How a Kalman filter works, in pictures | Bzarg.

[2] Gustafsson, F., Gunnarsson, F., Bergman, N., Forssell, U., Jansson, J., Karlsson, R., & Nordlund, P.-J. (2009). Particle Filters for Positioning, Navigation and Tracking. IEEE Transactions on Signal Processing, 50(2), 425–437. doi: 10.1109/78.978396

[3] Marín, D. A. A. (2012, August 14). Particle filter tutorial - File Exchange - MATLAB Central. Retrieved November 7, 2019, from https://www.mathworks.com/matlabcentral/fileexchange/35468-particle-filter-tutorial.

[4] [Srini, 2019] Srini, S. (2019). Particle Filter : A hero in the world of Non-Linearity and Non-Gaussian.