Tracking of the Drone in Space

Chaojie Wang

07/15/2015

1 Introduction

In this report, I want to present a way to track the position of our drone in space. I propose this to be the first task we need to complete. Since a object in space has 6 degrees of freedom, so there are 6 parameters we need to keep track of in order to fully specify the location of our drone. The idea is inspired from a paper, which I will include at the end as reference.

2 Degrees of Freedom and Coordinate

First of all, two concept are important for tracking the dorne: degrees of freedom and coordinate.

There are 3 coordinate system we care about. First one is body frame, which is always align with the drone itself. The reason why we need to think about body frame is that the data getting from the MPU-9150 is based on it. The second one is local frame, which doesn't rotate but moves with the drone. Local frame is useful as reference to measure angles. The third one is inertia frame, which is fixed and is treated as ultimate reference as where the drone is.

Degrees of freedom is all variables that can affect the outcome. In our case, there are six degrees of freedom: 3 parameters to specify how much the drone moves in space, and other 3 parameters to specify how much the drone rotates.

If you have doubt, you can consult the paper I include at the end. I think it does good job on explaining these ideas.

3 Tracking

If we have local frame and inertial frame. we can meanfully talk about the position and ratation of our drone. In other words, 6 degrees of freedom can be uniquely quantify. Let's define these 6 degrees as x, y, z, ϕ, θ and ψ. ϕ is with respect to x; θ is with respect to y; ψ is with respect to z. These 6 data and deritatives of these data(velocities and angular velocities) is important and will be stored in our microcontroller RAM.

Now let's specify what is our inertia frame and local frame. I propose that wherever the drone start as the inertia frame and local frame, for simplicity, with the apsumtion that the drone is on a level ground and staitc. This is simple, because it means x, y, z, ϕ, θ, ψ, their derivatives(velocities), $\dot{x}$, $\dot{y}$, $\dot{z}$, $\dot{\phi}$, $\dot{\theta}$, $\dot{\psi}$ and accelerations $\ddot{x}$, $\ddot{y}$, $\ddot{z}$, $\ddot{\phi}$, $\ddot{\theta}$, $\ddot{\psi}$are all 0 initially.

All these 12 data should be updated constantly while the drone is flying. this is done by following way. First of all, update ϕ, θ and ψ using information about its angular velocity and angular acceleration which is stored in RAM. For example ϕ:

$$\phi = \phi + \dot{\phi}\Delta t + \frac{1}{2}\ddot{\phi}\Delta t^2$$

Then update x, y, z with the velocity which is also stored in RAM, using simple simple motion equation. For example:

$$x = x + \dot{x}\Delta t + \frac{1}{2}\ddot{x}\Delta t^2$$

Next step is update $\dot{\phi}$, $\dot{\theta}$, $\dot{\psi}$. Angular velocity is easy, which is kinda the same as the x. For example, to $\dot{\phi}$ with the information about $\dot{\phi}$ which is from MPU-9150:

$$\dot{\phi} = \dot{\phi} + \ddot{\phi}\Delta t$$

Finally, we need to update $\dot{x}$, $\dot{y}$, $\dot{z}$. To get update this, we need accelerations. But the accelerations we get from MPU-9150 is with respect to body frame and gravity is one component. We can fix by using the matrix $[BtoL]$[1]. Then we should correct it in order to get the true acceleration by: invert the reading(add minus add in the , then add the gravitity. After we have the acceleration with respect to local frame, we can update the volecity like we did to other data. we denote by:

$$\vec{v} = \vec{v} + [BtoL]\vec{a}\Delta t$$

Lastly, the accelerations are updated for next time. The update simply means get the data fromt the sensor and clean them up.

Note that, any translational movement related vectors should be corrected, due to the particularity of data sensor for measuring acceleration. the correction formula is:

$$\vec{a} = -\vec{a}_r + \vec{g}$$

where $\vec{a}$ is the true vector, $\vec{a}_r$ is the data from the sensor, and $\vec{g}$ is the grativity vector.

All these update should be done in a small time interval. Our microcontroller should be quick enough.

4 Last Note

To fully understand these, one needs to undertand degrees of freedom and different coordinate system. consult the paper.

The equations above assume that acceleration when we get the data doesn't change within ΔT. And combine with simple physics about motion and rotation, the equation can be derived.

References

[1] Thomas S. Alderete *Simulator Aero Model Implementation*