Drone Sim

# Quadrotor in 2D: The Birotor

## Initial Exploration

In Physics class, before we try to do anything in 3D, we always started with single dimensions and added more later. A drone in 1D is fairly straightforward: You have a point mass or a line, it has some means of propulsion (must be a rocket, since there’s no way to pass “through” anything in 1D), and it runs its motor in response to gravity to maintain its position. This is pretty straightforward; after the initial run-up of the engine, and probably some second-order oscillating around the target spot, you reach an equilibrium of the motor firing at 1g to keep the point mass or line at the desired position. 1D drones are trivial.

So to have an interesting problem at all, we have to start with a 2D drone. I intend to model this as a rod suspended between two “propellers” (lines that generate perpendicular thrust), with some payload centered at the center of mass of the system. I’ll assume the width of the body-rod and payload are negligible for the sake of aerodynamics and rotational kinematics, and probably will begin developing this idea with the notion that either the payload mass is distributed along the body or that the mass of the body is negligible. This should really only affect the response function that the controller applies to the motors, not much interesting about the fundamental equations of motion or the methods in general.

The system itself is pretty straightforward. The force of the rotors is always perpendicular to the body, which is assumed to be rigid. Thus, we only need a single angle to represent the rotation of the system to calculate the force the rotors apply in the x and y dimensions. This angle even factors into the drag on the system from the air. Including air drag in the model at the outset ought to reduce the risk of perturbations sending the system wildly off to infinity. Likewise, limiting the force of the motors ought to keep the model from flying off, too.

As the photo of my notebook points out, I intend to decide the maximum force output of the rotors (probably something sane like 1mg or 2mg per rotor) and use that to calculate a Reynolds number based on a maximum velocity in the x direction (the drone at a 90-degree angle at max thrust), rather than trying to base it off of some real Reynolds number that approximates my system. I haven’t bothered to look up the real drag equation yet, just scribbled it from memory. It’s probably wrong; it certainly has weird dimensions. I do remember it going as velocity squared, though, and depending on the area orthogonal to the direction of motion (or rather, the length of the system in the y dimension, for this 2D model). It’s the best I could think of without really researching it. My rotational inertia factors are also from memory and probably wrong for a rod, too.

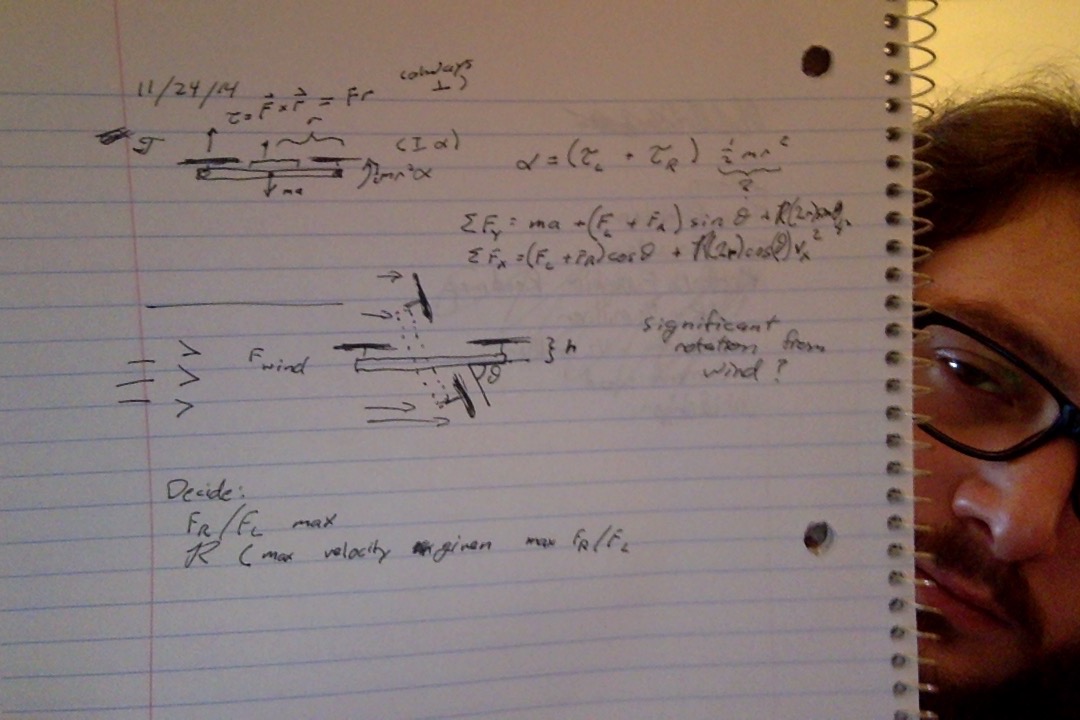


Figure - Notebook doodles, possibly some kinematics.

I tried to think of a way to add rotational perturbations from the wind, but I’m not sure they will be significant. I’m not even sure whether there is significant rotational (rather than just directional) perturbation from wind in real drones. For now, I’ll leave it out of the exploration.

I intend to try producing the model in python as a numerical, discrete-time-step simulation. To start off with, I’ll have a representation of the initial conditions of the system that amounts to initial thrust of the rotors and mass of the system. For starters, I’ll just hold the initial position and velocity and the mass and length of the system constant. Gravity is idealized--earth gravity at sea level, like all the Freshman physics books.

The first step will be to make sure the thing drops like a rock (literally) with no thrust. I’ll plan on only dealing with air resistance in the x direction, since the y has gravity to make it interesting and we intend mostly to hold altitude.

## Building the classes

I intend to try building the simulator in python. I’ll have classes for the drone model, for the environment, and for the simulation itself. The simulation will have one environment and as many drones as requested.

## The Initial Physics System Built

So far I have built the basic physics system (minus wind resistance) and started playing with varying the thrust. I thought I could get the drone to move back and forth, but it just crashes. I started off with a simulation to verify that the drone falls straight to the ground as expected with only gravity and no thrust.

## Online course on drone control systems

Now that I have a model, I figured it would be useful to learn about the command and control aspect, which would be the ultimate goal here. I found a pretty awesome-looking course online: <https://courses.edx.org/courses/TUMx/AUTONAVx/2T2014/info>

Unfortunately, installing the Robot Operating System (ROS) to do the homework is a long and tenuous process. So I’m working on getting that up and running so I can see what the course can teach me.