EE798T Course Project
Automatic Antenna Tracking System

Pratyush Gupta [200717] Anshuman Barnwal [200259]

1 Introduction

The problem statement is to design a ground station antenna tracking system that dynamically points a directed antenna at a moving UAV. Assuming that the drone is friendly, the antenna would also be receiving the kinematics of the UAV from some of the packets that it receives. In this article, we will describe our solution to this problem, challenges faced, and how we resolved those challenges.

2 Methodology

2.1 Control Strategy for the Antenna

The antenna is already present and constructed by the SPiN Labs. The antenna has a bi-modal actuator setup, one controlling azimuth ϕ and the other elevation θ angle. Given some reference angles ϕ^r, θ^r The current control setup is of a bang-bang controller, which directs the motors to turn by a accelerated ramp, followed by constant velocity and a deceleration ramp, the response decided as an open loop controller's output, but comparison happens at every step, making it a feedback setup. Also ϕ controller works first followed by θ controller.

We propose to the following:

1. Introduce more advanced and optimal control algorithms for individual axis control.
2. Couple both controllers to get a faster response by changing both angles simultaneously.
3. Develop a method to predict the response time given the current orientation (ϕ, θ) and the desired orientation (ϕ^r, θ^r), say Δt_a.
4. Analyse the path traced by the antenna top over the sphere S_a.

Note that the path traced on sphere S_a may not necessarily be a the shortest line between the two orientations on the sphere spanned by the antenna.

This phase of the project would require only working with the antenna structure/stand, and the reference can be provided directly without the need of a UAV.

2.2 Prediction for Correct Reference Angles

Let the kinematics data received at time t from the UAV be $D_t = (p_t, v_t, a_t)$. Let the current orientation of the antenna be defined using ϕ_t, θ_t, where ϕ_t is the azimuth angle and θ_t is the elevation angle at time t. The next packet would arrive at a dynamic rate (due to possible packet loss), but with an expected rate of $1 - 2$ Hz, denote by R_D.

Due to a relatively low feedback rate of pose, we aspire to take the advantage of higher order movement data of the UAV to create kinematics models that would predict the future pose of the UAV based on the history and the last seen packet data, at some timestep $t + \Delta t$ for any Δt.

We would also require to model the current rate and predict the arrival time of the next packet, denote by Δt_p.

Consider two phases, the first is when the antenna structure has to align to the initial pose of the UAV, at $t = 0$, where it's orientation might be quite deviated from the required orientation. In this case, $\Delta t_a > \Delta t_p$. This would lead us to update our reference as soon as the packet arrives, therefore cancelling the ongoing stand movement. The other case would be during flight, when the current orientation is near the reference, leading to $\Delta t_a < \Delta t_p$. The ideal scenario would be for the antenna to follow the predicted trajectory of the UAV, while reaching the final position at time $t + \Delta t_p$. This would essentially lead to the antenna being directed to the estimated position of the UAV at all times, where the estimate is itself corrected on average every $1/R_D$ `s`. The above can be achieved by many possible ways, one of which would be to break down the trajectory itself sequences of waypoints. The waypoints would be such that they are equally spaced over the sphere S_a, number of waypoints can be kept a constant, which would be small in number because Δt_p is nearly 500`ms`.

If we treat waypoints as constant length lines over S_a, we would not need to predict Δt_p. This is because in both cases of cold start and mid-flight, we can keep on tracing the sphere with constant length lines by giving waypoints, until the next packet arrives.

We would need to do the following:

1. Setup the scripts that intimate kinematics data as IP packets between the onboard computer and the antenna controller.
2. Work out an optimal model based on current and past kinematics of the UAV that would predict the pose at $t + \Delta t$, let $P(t)$, which can be directly translated to antenna orientation $P_a(t)$.
3. Simulate a UAV with its pose being broadcasted on a computer and tune the above

We will discuss our solutions to the objectives stated in the last two solutions now. First, we will consider the three objectives we have just stated.

3 Prediction for Correct Reference Angles

The pipeline works in these main steps.

3.1 Getting the data from the drone

We base our system on ROS. So, we operate the drone's computer and the antenna computer on the same ROS Network. There are two cases here

3.1.1 Using the MoCap

In the indoor setting, we can use the Qualisys Motion Capture System. The system provides the position and velocity of the drone. In our case the information is available on the topic $qualisys\backslash AAT\backslash odom$

3.1.2 Outdoor Setting

Thee drone still publishes information about the position and velocity that it obtains from the onboard GPS. Since the system also uses MAVROS, we can access this information on the topic $\backslash mavros\backslash local\_postion\backslash pose$

In both the cases, the information about the acceleration will come from the drone's IMU. The topic is $\backslash mavros\backslash imu\backslash data$

3.2 The drone position predictor

The inputs for this predictor are :

- Position p
- Velocity v
- Acceleration a

Then we set a parameter t_h, the time horizon. At a time T, We predict the position of the UAV at the time $T + t_h$. We do this using a simple formula

$$p_{T+t_h} = p_T + vt_h + \frac{1}{2}at_h^2$$

This formula is essentially a termination of the Taylor series expansion for the function $x = f(t)$. We do this due to the unavailability of higher derivatives beyond acceleration. This formula can also be executed very quickly.

3.2.1 Dealing with lag

Once source of possible error in prediction comes from the time it may take for the UAV to send the required information to the antenna. However, the UAV sends its information with a timestamp. Thus when we attempt to make a prediction, we know the exact delay. To compensate for the lag, we make the prediction at $t = T + t_h + lag$.

3.2.2 Conversion to Polar Coordinates

The predicted position is in cartesian coordinates. We convert it to the Polar Coordinates.

3.2.3 Conversion to motor steps

Let the position be (r, θ, ϕ) in polar coordinates. For the antenna control problem r is not relevant. We only care about θ and ϕ. The conversion from degrees to motor steps is a linear transformation :

$$steps = A\theta + B$$

where we determine θ experimentally. The other angle, ϕ, behaves the same way.

3.2.4 Computing the motor inputs

The problem now reduces to computing a command C such that the motor reaches the desired angle at time $T + t_h$. We will describe the control system for this in the next section.

3.3 Simulating the drone

Let x denote the three dimensional position of the drone. Then we simulate the drone along trajectories

$$x = f(t)$$

We ensure that this function f is at least twice differentiable so that we get nice expressions for the velocity and acceleration. An example of such a trajectory is a circle.

4 Control System

4.1 Computation of Motor Characteristics

We began by computing the motor characteristics. Formally, we wanted to look at the function

$$\theta = f(t, \theta_{goal})$$

where θ is the current angle of the motor, and θ_{goal} is the destination that we asked the motor to reach. The time is represented by t. We performed a number of experiments to understand this function. An example of this graph is shown below.

We observe a few things about the function f

- The angular velocity of the motor is limited between $[\omega_1, \omega_2]$
- The angular acceleration of the motor takes only two values α_1 and α_2, where $\alpha_1 > 0$ and $\alpha_2 < 0$

If the quantities $\omega_1, \omega_2, \alpha_1, \alpha_2$ are known, then we can completely define the function f. Experimentally we obtained the following values for the motors:
θ motor

- ω_1
- ω_2
- α_1
- α_2

ϕ motor

- ω_1
- ω_2
- α_1
- α_2

4.2 Finding these quantities

Let us say we have a plot that looks like the following. This is a fairly typical image.
By computing the slope of the upward sloping line we get α_1
By computing the slope of the downward sloping line we get α_2
We can directly see the saturation velocities ω_1 and ω_2 from the plot.

4.3 Computing nice motor inputs

Let us say we need to reach a certain $\theta_{desired}$ in time $t_{horizon}$. If we simply give the command $\theta_{desired}$ to the motor, it will reach $\theta_{desired}$ according to the function f defined earlier. The time it takes, need not be equal to $t_{horizon}$. Thus we need to compute such an angle θ_{goal}, such that the motor reaches angle $\theta_{desired}$ at time $t_{horizon}$ when given the command θ_{goal}. Formally

$$\theta_{desired} = f(t_{horizon}, \theta_{goal})$$

Since we can completely describe the function f, this can be computed quickly.