Tracking a Moving Target with a Quadrotor UAV

Samuel Ubellacker, Henry Hu

***Abstract*— We investigate the problem of tracking a moving ground vehicle with a flying quadrotor. We simulate the two vehicles with noisy sensor inputs passed to the quadrotor at a reduced rate to simulate an on-board camera. The current and future states of the ground vehicle are estimated using a combination of the Extended Kalman Filter (EKF) and polynomial fitting. Using the ground vehicle's predicted trajectory as a reference trajectory, we then use a linearized dynamics of the quadrotor to calculate an optimal trajectory that allows the quadrotor to match the position and velocity of the tracked car. The calculation of the quadrotor trajectory uses linear model predictive control (LMPC) with actuator and state contraints. We evaluate our results by showing the prediction error of our state estimation techniques and the convergence of the quadrotor control algorithm over time.**

I. Disclaimer

Samuel Ubellacker is using this work as a preliminary baseline for his research on grasping moving targets with a soft aerial platform.

II. Introduction

Quadrotor vehicles are widely used and studied as a simple flight control system. Due to their small size and flight range, however, it can be useful to have quadrotors interact with ground vehicles in various manners, without the aid of external sensors (such as external motion capture). We investigate the tracking of a moving ground vehicle with a quadrotor.

Our problem can be viewed as two sub-problems which we solved one-by-one, in simulation only. First, the quadrotor's sensor data can be noisy and come in at a lower rate than its processing capabilities, because the data comes from an on-board camera rather than an external sensor system. To simulate noisy and sparse input data, we add Gaussian noise to the ground-truth location of the car and only make it available to our calculations at a slower rate than the actual processing rate of the vehicles. We use an Extended Kalman Filter (EKF) and polynomial fitting to obtain an estimate of the ground vehicle's state and anticipated trajectory. Second, given an estimate of the car's trajectory, we need to plan an optimal route to achieve the same velocity and position (up to a constant offset). Because the ground vehicle is dynamic, we choose to use linear model predictive control (LMPC), which is able to both anticipate future states of the target and quickly replan trajectories when new information is obtained.

The problem of tracking a vehicle with a quadrotor is interesting to our group because it is an advanced implementation of the underactuated techniques that we have studied in class. In particular, we learned a lot by implementing MPC from scratch, including the linearization of the dynamics. The additional layers of complexity in our problem, which involved two types of interacting vehicles, also made it interesting to study and simulate.

Tracking a target vehicle using MPC is one possible foundation for learning to dynamically grasp moving objects, which is the thesis work of a group member. Making a quadrotor follow a ground vehicle is a versatile task and has other interesting applications as well, including charging the battery of the quadrotor.

S. Ubellacker, H. Hu are with the Dept. of Electrical Engineering and Computer Science, Massachusetts Institute of Technology, Cambridge, MA, subella@mit.edu, henryhu@mit.edu

III. Target State Estimation

On-board perception and object detection pose several challenges in regards to state estimation of the moving target. If the target happens to move out of the field of view (FOV) of our camera, then it no longer becomes possible to estimate its state. Additionally, sensor noise degrades the integrity of our measurements, which could cause our control policy to fail. The refresh rate of a camera sensor is typically much slower than that of the control loop, thus motivating the need for state prediction between updates.

This work addresses the issues of sensor noise and target prediction, and leaves the FOV considerations for future work. In practice, quadrotor's typically use high FOV cameras or 360 degree FOV cameras, which eliminates this constraint entirely. Specifically, an Extended Kalman filter is used for state estimation and a polynomial fitting method is used for state prediction.

A. Ground Vehicle Dynamics

In discrete time, the dynamics of the moving ground vehicle can be written as:

$$\mathbf{x}(k+1) = f(\mathbf{x}(k), \mathbf{u}(k)) + \mathbf{w}(k) \tag{1}$$

with $\mathbf{x}$, $\mathbf{u}$, $\mathbf{w}$ representing the systems state, control input, and process noise respectively.

$$\begin{cases} \mathbf{x} = \begin{bmatrix} x & y & z & \psi & v \end{bmatrix}^T \\ \mathbf{u} = \begin{bmatrix} u_1 & u_2 \end{bmatrix}^T \end{cases} \tag{2}$$

where x, y, z represent the world coordinates in an NED reference frame, ψ represents the yaw, or angle with respect to the x-axis, and v represents the tangential velocity of the vehicle. The process noise $\mathbf{w}$, is assumed to be multivariate Gaussian with zero-mean and covariance $\mathbf{Q}_k$.

The target vehicle follows standard vehicle dynamics based on non-holonomic movement constraints, for a discrete timestep Δt:

$$f(\mathbf{x},\mathbf{u}) = \begin{bmatrix} x + v\cos(\psi)\Delta t \\ y + v\sin(\psi)\Delta t \\ z \\ \psi + u_1\Delta t \\ v + u_2\Delta t \end{bmatrix} \tag{3}$$

As seen in later sections, it is convenient to re-express the vehicle state and corresponding output in $\mathbb{R}^{12}$ with complete cartesian coordinates and euler angles $\mathbf{r}$:

$$\begin{aligned} \mathbf{x} &= \begin{bmatrix} x & y & z & \dot{x} & \dot{y} & \dot{z} & \phi & \theta & \psi & \dot{\phi} & \dot{\theta} & \dot{\psi} \end{bmatrix}^T \\ \mathbf{r} &= C\mathbf{x} \end{aligned} \tag{4}$$

where $\theta = \dot{\theta} = \phi = \dot{\phi} = 0$ as the ground vehicle is assumed to always lie flat on the floor.

B. Extended Kalman Filter

The EKF consists of a *prediction* step, where the transition model is used to estimate the current state and uncertainty, and an *correction* step which updates the estimate when sensor measurements are made.

Prediction Step. The prediction $\hat{\mathbf{x}}_{k|k}$ and error covariance $\mathbf{P}_{k|k}$ at time k given all the previous states can be written as:

$$\begin{aligned} \hat{\mathbf{x}}_{k|k} &= f(\hat{\mathbf{x}}_{k|k-1}, \mathbf{u}_{k-1}) \\ \mathbf{P}_{k|k-1} &= \mathbf{A}_k \mathbf{P}_{k-1|k-1} \mathbf{A}_k^T + \mathbf{Q}_k \end{aligned} \tag{5}$$

where $\mathbf{A}_k$ represents the ground vehicle's linearized transition matrix obtained through the Jacobian of f:

$$\mathbf{A}_k = \begin{bmatrix} 1 & 0 & 0 & -v\sin(\psi)\Delta t & \cos(\psi)dt \\ 0 & 1 & 0 & v\cos(\psi)\Delta t & \sin(\psi)dt \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix} \tag{6}$$

Correction Step. The predicted state is updated when a sensor measurement $\mathbf{z}_k$ is available. We assume camera sensor is capable of measuring the position and heading of the target, but not its velocity v. The observation model can be written as:

$$\mathbf{z}_k = \mathbf{H}_k \mathbf{x}_k + \mathbf{v}_k \tag{7}$$

where $\mathbf{H}_k$ represents the measurement matrix given by:

$$\mathbf{H}_k = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \end{bmatrix} \tag{8}$$

and the measurement noise $\mathbf{v}_k$, is assumed to be multivariate Gaussian with zero mean and covariance $\mathbf{R}_k$.

The measurement residual can then be computed as:

$$\tilde{\mathbf{y}}_k = \mathbf{z}_k - \mathbf{H}_k \hat{\mathbf{x}}_{k|k-1} \tag{9}$$

The residual covariance $\mathbf{S}_k$ and Kalman gain $\mathbf{K}_k$ are given as:

$$\begin{aligned} \mathbf{S}_k &= \mathbf{H}_k \mathbf{P}_{k|k-1} \mathbf{H}_k^T + \mathbf{R}_k \\ \mathbf{K}_k &= \mathbf{P}_{k|k-1} \mathbf{H}_k^T \mathbf{S}_k^{-1} \end{aligned} \tag{10}$$

The state prediction and error covariance can then be updated through the Kalman gain $\mathbf{K}_k$:

$$\begin{aligned} \hat{\mathbf{x}}_{k|k} &= \hat{\mathbf{x}}_{k|k-1} + \mathbf{K}_k \tilde{\mathbf{y}}_k \\ \mathbf{P}_{k|k} &= (I - \mathbf{K}_k \mathbf{H}_k) \mathbf{P}_{k|k-1} \end{aligned} \tag{11}$$

C. Regression

We fit a polynomial to the previous L states provided by the EKF. This polynomial allows us to model the trajectory of the ground vehicle and predict its future motion. We consider the following regularized polynomial fitting problem:

$$\min_{\mathcal{P}} \sum_{i=0}^{L} \|\mathcal{P}(t_i) - p_i\|^2 + L\lambda \int_{t_l}^{t_m} \left\|\mathcal{P}^{(r)}(t)\right\|^2 dt \tag{12}$$

where we want to compute a polynomial $\mathcal{P}(t) \doteq c_o + c_1 t + \ldots + c_n t^n$ of order n given noisy scalar observations $p_0, \ldots, p_L \in \mathbb{R}$ at given times $t_0, \ldots, t_L \in \mathbb{R}$; the optimization fits the data $p_0, \ldots, p_L$ while adding a regularization on the integral of the squared norm of the r-th derivative $\mathcal{P}^{(r)}(t)$ of the polynomial. The amount of regularization is controlled by the user-specified constant $\lambda \geq 0$. We choose to penalize the 2nd derivative, making the assumption that the vehicle does not change acceleration drastically. The above can be solved in a closed form, as in [4]

IV. Quadrotor Model Predictive Control

Model Predictive Control is a form of optimal control which computes the current control action by considering the evolution of the system over a finite time horizon. The optimization can be formulated as a numerical program with an objective function which includes the projected state and planned future control actions over the time horizon.

Unlike *reactive* algorithms, such as a PID, the *proactive* nature of MPC makes it well suited for tracking problems where the reference state is constantly changing. Additionally, MPC schemes benefit from a constrained formulation allowing for safety critical constraints such as velocity or actuator limits to be easily incorporated into the optimization.

In Linear Model Predictive Control (LMPC), the quadrotor dynamics must linearized around an operating point, while in Nonlinear Model Predictive Control (NMPC) one can directly use the nonlinear dynamics. NMPC designs can perform highly aggressive maneuvers in the full regime of the nonlinear dynamics, while the operating region of LMPC designs is much smaller. However, LMPC designs are guaranteed to be convex and can be solved with Quadratic Programming (QP). NMPC designs, on the other hand, are possibly nonconvex and challenging to solve numerically.

In this work, we operate under conditions in which the use of LMPC is justified. The target trajectory is assumed to be a straight line or have a slight curvature with no aggressive maneuvers. Consequently, the drone's dynamics evolves along an approximately straight line with constant orientation. It is shown in [8], that the performance of LMPC and NMPC under these conditions is similar. In future work, a control scheme that exploits the full dynamics of the quadrotor will be necessary for performing aggressive tracking maneuvers.

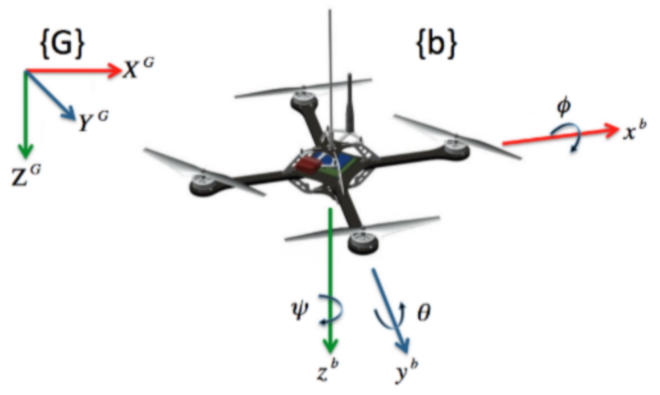

Fig. 1: Dynamical model of quadrotor platform

A. *Quadrotor Dynamics*

As is common in aerial vehicles, we express the quadrotor dynamics in an NED (North-East-Down) coordinate system, with a moving frame attached to the body of the quadrotor and a fixed inertial frame (Fig. 1). We express the state of the quadrotor as:

$$\mathbf{x} = \begin{bmatrix} x & y & z & \dot{x} & \dot{y} & \dot{z} & \phi & \theta & \psi & \dot{\phi} & \dot{\theta} & \dot{\psi} \end{bmatrix}^T \quad (13)$$

where x, y, z, and $\dot{x}$, $\dot{y}$, $\dot{z}$ represent the position and velocity, respectively, of the quadrotor with respect to the fixed inertial frame. The orientation of the quadrotor with respect to the fixed frame is expressed in euler angles ϕ, θ, ψ, with corresponding euler rates $\dot{\phi}$, $\dot{\theta}$, $\dot{\psi}$.

It is shown in [11] that the quadrotor dynamics can be written in state space form as:

$$\begin{cases} \dot{x}_1 = \dot{x} = x_2 \\ \dot{y}_1 = \dot{y} = y_2 \\ \dot{z}_1 = \dot{z} = z_2 \\ \dot{x}_2 = \ddot{x} = -\frac{F_z}{m}(s\phi_1 c\psi_1) \ + c\phi_1 c\psi_1 s\theta_1) \\ \dot{y}_2 = \ddot{y} = -\frac{F_z}{m}(c\phi_1 s\psi_1 s\theta_1 \ - c\psi_1 s\phi_1) \\ \dot{z}_2 = \ddot{z} = g - \frac{F_z}{m}(c\phi_1 c\theta_1) \\ \dot{\phi}_1 = \dot{\phi} = \phi_2 \\ \dot{\theta}_1 = \dot{\theta} = \theta_2 \\ \dot{\psi}_1 = \dot{\psi} = \psi_2 \\ \dot{\phi}_2 = \frac{I_y - I_z}{I_x}\theta_2\psi_2 + \frac{\tau_x}{I_x} \\ \dot{\theta}_2 = \frac{I_z - I_x}{I_y}\phi_2\psi_2 + \frac{\tau_y}{I_y} \\ \dot{\psi}_2 = \frac{I_x - I_y}{I_z}\phi_2\theta_2 + \frac{\tau_z}{I_z} \end{cases} \quad (14)$$

where s and c denote sin and cos; F_z represents the upwards force aligned with the z-body axis of the quadrotor; τ_x, τ_y, τ_z, represent the torques applied to each body axis; and I_x, I_y, I_z are the principle moments of inertia of the quadrotor.

The dynamics of the whole system in vector form can then be written as:

$$\dot{\mathbf{x}} = f(\mathbf{x}) + g(\mathbf{x})\mathbf{u} \quad (15)$$

with:

$$f(\mathbf{x}) = \begin{bmatrix} x_2 \\ y_2 \\ z_2 \\ 0 \\ 0 \\ g \\ \phi_2 \\ \theta_2 \\ \psi_2 \\ \frac{I_y - I_z}{I_x}\theta_2\psi_2 + \frac{\tau_x}{I_x} \\ \frac{I_z - I_x}{I_y}\phi_2\psi_2 + \frac{\tau_y}{I_y} \\ \frac{I_x - I_y}{I_z}\phi_2\theta_2 + \frac{\tau_z}{I_z} \end{bmatrix}$$

$$g(\mathbf{x}) = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ -\frac{1}{m}(s\phi_1 s\psi_1 + c\phi_1 c\psi_1 s\theta_1 & 0 & 0 & 0 \\ -\frac{1}{m}(c\phi_1 s\psi_1 s\theta_1 - c\psi_1 s\phi_1) & 0 & 0 & 0 \\ -\frac{1}{m}(c\phi_1 c\theta_1) & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & \frac{1}{I_x} & 0 & 0 \\ 0 & 0 & \frac{1}{I_y} & 0 \\ 0 & 0 & 0 & \frac{1}{I_z} \end{bmatrix} \quad (16)$$

$$\mathbf{u} = \begin{bmatrix} F_z & \tau_x & \tau_y & \tau_z \end{bmatrix}^T$$

We seek to linearize the above equation for use in the linear model predictive control algorithm described in the following section. The nonlinear dynamics are linearized around the near-hover equilibrium point $\bar{\mathbf{x}}, \bar{\mathbf{u}}$ determined by evaluating the system at rest, when $\dot{\bar{\mathbf{x}}} = 0$:

$$0 = f(\bar{\mathbf{x}}) + g(\bar{\mathbf{x}})\bar{\mathbf{u}} \quad (17)$$

We operate in a region where the angles ϕ, θ are sufficiently small such that small angle approximations hold. Solving equation 17 leads to equilibrium at points of the form:

$$\begin{cases} \bar{\mathbf{x}} = \begin{bmatrix} \bar{x} & \bar{y} & \bar{z} & 0 & 0 & 0 & 0 & 0 & \bar{\psi} & 0 & 0 & 0 \end{bmatrix}^T \\ \bar{\mathbf{u}} = \begin{bmatrix} mg & 0 & 0 & 0 \end{bmatrix}^T \end{cases} \quad (18)$$

Evaluating the first order Taylor expansion at the above equilibrium yields:

$$\frac{\partial f(\mathbf{x})}{\partial \mathbf{x}}\bigg|_{\substack{\mathbf{x}=\bar{\mathbf{x}}\\ \mathbf{u}=\bar{\mathbf{u}}}} = A_c, \frac{\partial g(\mathbf{x})\mathbf{u}}{\partial \mathbf{u}}\bigg|_{\substack{\mathbf{x}=\bar{\mathbf{x}}\\ \mathbf{u}=\bar{\mathbf{u}}}} = B_c$$

$$A_c = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & -g & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & g & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \tag{19}$$

$$B_c = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ -\frac{1}{m} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & \frac{1}{I_x} & 0 & 0 \\ 0 & 0 & \frac{1}{I_y} & 0 \\ 0 & 0 & 0 & \frac{1}{I_z} \end{bmatrix}$$

The linearized continuous dynamics can now be written in the following form, with $\tilde{\mathbf{x}} = \mathbf{x} - \bar{\mathbf{x}}, \tilde{\mathbf{u}} = \mathbf{u} - \bar{\mathbf{u}}$:

$$\dot{\tilde{\mathbf{x}}} = A_c\tilde{\mathbf{x}} + B_c\tilde{\mathbf{u}} \tag{20}$$

The above can be discretized using a timestep Δt to obtain the linearized discrete time dynamics:

$$\tilde{\mathbf{x}}(k+1) = A\tilde{\mathbf{x}}(k) + B\tilde{\mathbf{u}}(k) \tag{21}$$

where A and B are the discrete analogs of A_c and B_c.

We can define the output vector $\tilde{\mathbf{y}}_{quad}(k)$ as:

$$\tilde{\mathbf{y}}_{quad}(k) = C\tilde{\mathbf{x}}(k)$$

$$C = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \tag{22}$$

B. Prediction

The future P states of the quadrotor can be estimated given the future M control inputs. Here, P and M are referred to as the *prediction horizon* and *control horizon*, respectively. Typically, the control horizon is selected such that $M < \frac{1}{2}P$. After the end of the control horizon, the remainder of the estimated states are calculated assuming a control input of $\mathbf{0}$. The consideration of additional timesteps allows for the full effect of the control action to be realized.

Let $\hat{\mathbf{x}}(k)$ represent the predicted state at time k using the discrete linearized dynamics. Then, the future P states can be expressed as:

$$\begin{aligned} \hat{\mathbf{x}}(k+1) &= A\mathbf{x}(k) + B\hat{\mathbf{u}}(k) \\ \hat{\mathbf{x}}(k+2) &= A(A\mathbf{x}(k) + B\hat{\mathbf{u}}(k)) + B\hat{\mathbf{u}}(k+1) \\ &= A^2\mathbf{x}(k) + AB\hat{\mathbf{u}}(k) + B\hat{\mathbf{u}}(k+1) \\ &\vdots \\ \hat{\mathbf{x}}(k+P) &= A^P\mathbf{x}(k) + \\ &\quad A^{P-M}\sum_{i=1}^{P} A^{i-1}B\hat{\mathbf{u}}(k+M-i) \end{aligned} \tag{23}$$

The future predicted quadrotor states and output can be written succinctly in the following form:

$$\hat{\mathbf{x}}_k = \mathbf{A}_k\mathbf{x}(k) + \mathbf{B}_k\hat{\mathbf{u}}_k \tag{24}$$

with:

$$\hat{\mathbf{x}}_k = \begin{bmatrix} \hat{x}(k+1) \\ \hat{x}(k+2) \\ \vdots \\ \hat{x}(k+P) \end{bmatrix}$$

$$\mathbf{A}_k = \begin{bmatrix} A \\ A^2 \\ \vdots \\ A^P \end{bmatrix}$$

$$\mathbf{B}_k = \begin{bmatrix} B & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ A^{M-1}B & A^{M-2}B & \cdots & B \\ A^M B & A^{M-1}B & \cdots & AB \\ \vdots & \vdots & \ddots & \vdots \\ A^{P-1}B & A^{P-2}B & \cdots & A^{P-M}B \end{bmatrix} \tag{25}$$

$$\hat{\mathbf{u}}_k = \begin{bmatrix} \hat{u}(k) \\ \hat{u}(k+1) \\ \vdots \\ \hat{u}(k+M) \end{bmatrix}$$

The output can then be written as:

$$\hat{\mathbf{y}}_k = \begin{bmatrix} C\hat{x}(k+1) \\ C\hat{x}(k+2) \\ \vdots \\ C\hat{x}(k+P) \end{bmatrix} \tag{26}$$

A reference trajectory $\hat{\mathbf{r}}_k$ can be obtained by sampling the target's future P states using the fitted polynomial described in section III-C.

We seek the value for $\hat{\mathbf{u}}_k$ that minimizes the difference between the predicted quadrotor's output $\hat{\mathbf{y}}_k$ and the target's output $\hat{\mathbf{r}}_k$, such that the quadrotor stays in a safe operating

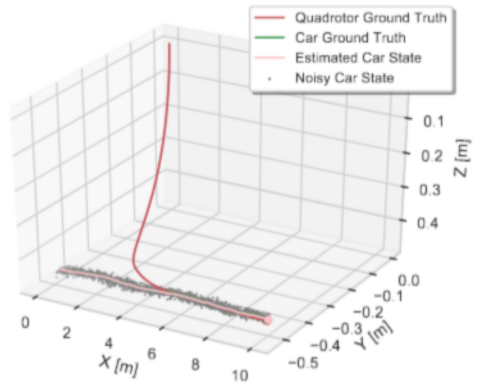

Fig. 2: An overview of the moving tracking problem. The red and green lines represent the quadrotor and target setpoint trajectories, respectively. The pink line represents the EKF state estimate; the gray dots are the observed states.

region $(\mathbb{U}, \mathbb{X})$. Assume that the safe operating regions can be expressed by linear constraints on the control input and state. The problem can be formulated as a quadratic program:

$$\begin{aligned} J = \min_{u} \, & (\hat{\mathbf{y}}_k - \hat{\mathbf{r}}_k)^T Q (\hat{\mathbf{y}}_k - \hat{\mathbf{r}}_k) + \hat{\mathbf{u}}_k^T R \hat{\mathbf{u}}_k \\ & \text{subject to:} \\ & \quad \hat{\mathbf{u}}_k \in \mathbb{U}^{M+1} \\ & \quad \hat{\mathbf{x}}_k \in \mathbb{X}^{P} \end{aligned} \tag{27}$$

where Q and R consist of weights for the elements of the output and actuator vectors, respectively, at every time step. Substituting (24) into (28) and removing constant terms:

$$\begin{aligned} J = \min_{u} \quad & \frac{1}{2} \hat{\mathbf{u}}_k^T (\mathbf{B}_k^T Q \mathbf{B}_k + R) \hat{\mathbf{u}}_k \\ & + ((\mathbf{A}_k \mathbf{x}(k) - \hat{\mathbf{r}}_k)^T Q \mathbf{B}_k) \hat{\mathbf{u}}_k \\ \text{subject to:} & \\ & \hat{\mathbf{u}}_k \in \mathbb{U}^{M+1} \\ & \hat{\mathbf{x}}_k \in \mathbb{X}^{P} \end{aligned} \tag{28}$$

Constraints on $\hat{\mathbf{x}}_k$ can be re-expressed as linear constraints on $\hat{\mathbf{u}}_k$ in the form $G\hat{\mathbf{u}}_k \le h$ by leveraging the linearized dynamics. For example, a constraint $z \le Z_f$ on the z-coordinate of the drone can be expressed as follows:

$$C_z \mathbf{B}_k \hat{\mathbf{u}}_k \le \begin{bmatrix} Z_F \\ \vdots \\ Z_F \end{bmatrix} - C_z \mathbf{A}_k \mathbf{x}(k) \tag{29}$$

where C_z is a matrix which selects the z coordinate of the state vectors at each time step.

V. Experiments

Our experiments show that, in simulation, the model predictive control converges to minimal tracking error for varying target velocities.

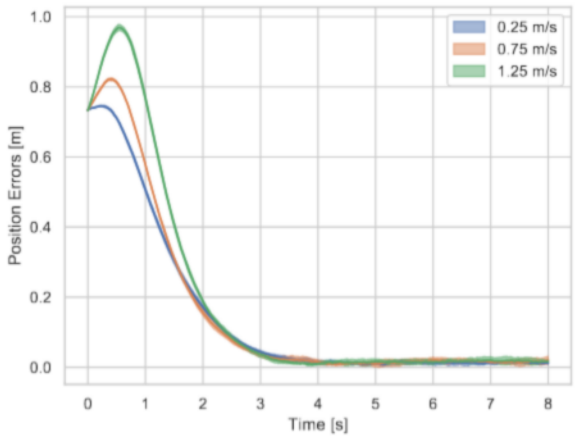

Fig. 3: Position tracking errors for the model predictive controller when tracking a moving target. Statistics are computed over 5 runs each with target velocities of $0.25\,\mathrm{m/s}$, $0.75\,\mathrm{m/s}$, and $1.25\,\mathrm{m/s}$.

A. Model Predictive Control Tracking Performance

Our experiment validates the tracking performance of linear model predictive control when the vehicle is assumed to move at a constant velocity and heading. We record data for 5 different trials each of velocities at $0.25\,\mathrm{m/s}$, $0.75\,\mathrm{m/s}$, and $1.25\,\mathrm{m/s}$. The target's process and sensor noise covariance was set to be $1 \times 10^{-6}\,\mathrm{I}$ and $1 \times 10^{-4}\,\mathrm{I}$, respectively, with I representing the identity matrix.

The controller quickly converges to the target's position even at higher velocities (Fig. 3). The steady state error is very small, even for higher velocities.

VI. Conclusion

We successfully implemented a system to track a moving ground vehicle with a flying quadrotor. We used a combination of EKF and polynomial fitting to perform state estimation on the ground vehicle, followed by LMPC to calculate the optimal quadrotor trajectory to reach the position and velocity of the ground vehicle. For future work, one team member will implement dynamic grasping of an object from a ground vehicle, which can be done using tracking with high precision. We also would be interested in further tuning the hyperparameters of the setup, including the control and prediction horizons in the LMPC.

References

[1] D. Falanga, A. Zanchettin, A. Simovic, J. Delmerico, and D. Scaramuzza, "Vision-based autonomous quadrotor landing on a moving platform," in *2017 IEEE International Symposium on Safety, Security and Rescue Robotics (SSRR)*, 2017, pp. 200–207.

[2] J. Thomas, J. Welde, G. Loianno, K. Daniilidis, and V. Kumar, "Autonomous flight for detection, localization, and tracking of moving targets with a small quadrotor," *IEEE Robotics and Automation Letters*, vol. 2, no. 3, pp. 1762–1769, 2017.

[3] D. Falanga, P. Foehn, P. Lu, and D. Scaramuzza, "Pampc: Perception-aware model predictive control for quadrotors," 04 2018.

[4] J. Chen, T. Liu, and S. Shen, "Tracking a moving target in cluttered environments using a quadrotor," in *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2016, pp. 446–453.

[5] A. Paris, B. T. Lopez, and J. P. How, "Dynamic landing of an autonomous quadrotor on a moving platform in turbulent wind conditions," *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 9577–9583, 2020.

[6] J. A. Macés-Hernández, F. Defaÿ, and C. Chauffaut, “Autonomous landing of an uav on a moving platform using model predictive control,” in *2017 11th Asian Control Conference (ASCC)*, 2017, pp. 2298–2303.

[7] Y. Feng, C. Zhang, S. Baek, S. Rawashdeh, and A. Mohammadi, “Autonomous landing of a uav on a moving platform using model predictive control,” *Drones*, vol. 2, p. 34, 10 2018.

[8] M. S. Kamel, M. Burri, and R. Siegwart, “Linear vs nonlinear mpc for trajectory tracking applied to rotary wing micro aerial vehicles,” vol. 50, 07 2017.

[9] D. Seborg, D. Mellichamp, T. Edgar, and F. Doyle, *Process Dynamics and Control*. John Wiley & Sons, 2010, pp. 386–410. [Online]. Available: https://books.google.com/books?id=_PQ42kOvtfwC

[10] M. Belkheiri, A. Rabhi, A. E. Hajjaji, and C. Pegard, “Different linearization control techniques for a quadrotor system,” in *CCCA12*, 2012, pp. 1–6.

[11] F. Sabatino, “Quadrotor control: modeling, nonlinear control design, and simulation,” 2015.