TRAJECTORY GENERATION AND CONTROL FOR QUADROTOR GRASPING AND PERCHING

JUSTIN THOMAS
GRASP Lab
Department of Mechanical Engineering and Applied Mechanics
University of Pennsylvania
Philadelphia, PA 19104
jut@seas.upenn.edu

ADVISOR: DR. VIJAY KUMAR

PH.D. QUALIFYING EXAMINATION

COMMITTEE

Dr. Vijay Kumar
Advisor

Dr. Katherine Kuchenbecker
Committee Chair

Dr. George Pappas
Math Examiner

May 17, 2012

Trajectory Generation and Control for Quadrotor Grasping and Perching

Justin Thomas

***Abstract*—When quadrotors are placed in real world environments, it will be desirable to perch or land as well as grasp objects reliably and in a timely manner. This paper explores and demonstrates a variety of methods that can be used for such planning and control. The differential flatness of a quadrotor is leveraged to enable planning in a simplified state space, which allows for algebraic and numerical methods for generating optimal trajectories. Additionally, three different controllers, including a non-linear controller on SE(3), will be discussed. The results of simulated controllers and real trajectories are presented.**

I. Introduction

From construction [1] to aerobatics [2], recent years have seen much improvement in the capabilities of Unmanned Aerial Vehicles (UAVs) [3]. Other than [4] and [5], however, there has been a lack of coverage regarding perching and grasping for quadrotors. These capabilities are closely related and can improve the usefulness of UAVs by enabling time and energy efficient interaction with the environment. For example, perching could be used to improve mission duration by reducing required flight time, it could be a way to safely outlast high winds, or it could allow for quick silence in stealth operations. Grasping enables manipulation of the environment and would facilitate the acquisition and transportation of objects. Overlapping both of these concepts is trajectory planning and the precise control necessary for quick and accurate tracking of a target object or location. The results presented in this paper utilized the GRASP Multiple Micro UAV Testbed [6] and leverage a motion capture system to accurately determine the state of the quadrotor [7].

II. Quadrotor Background

A. Dynamics

The quadrotor considered is produced by Ascending Technologies [8]. It is an unmanned micro aerial vehicle that has four coplanar rotors perpendicularly located a distance $l \in \mathbb{R}$ from the body center. One pair of counter-rotating propellers are used so that the net yaw moment is approximately zero at hover. The inertial frame, $\mathcal{A}$, with basis $\{\mathbf{a}_1, \mathbf{a}_2, \mathbf{a}_3\} \in \mathbb{R}^3$ and the body frame of the robot, $\mathcal{B}$, with basis $\{\mathbf{b}_1, \mathbf{b}_2, \mathbf{b}_3\} \in \mathbb{R}^3$ are defined with their third axes pointed upwards as depicted in Figure 2. The rotor speeds, $\omega_i \in \mathbb{R}$, are regulated to control the net thrust, $u_1 \in \mathbb{R}$, along the $\mathbf{b}_3$ axis and moments $u_2 \in \mathbb{R}$, $u_3 \in \mathbb{R}$, and $u_4 \in \mathbb{R}$ about the $\mathbf{b}_1$, $\mathbf{b}_2$, and $\mathbf{b}_3$ axes, respectively. The force, $F_i \in \mathbb{R}$, and moment, $M_i \in \mathbb{R}$, produced by each rotor is approximately related to the rotor speeds using $k_f \in \mathbb{R}$ and $k_m \in \mathbb{R}$ in the following equations

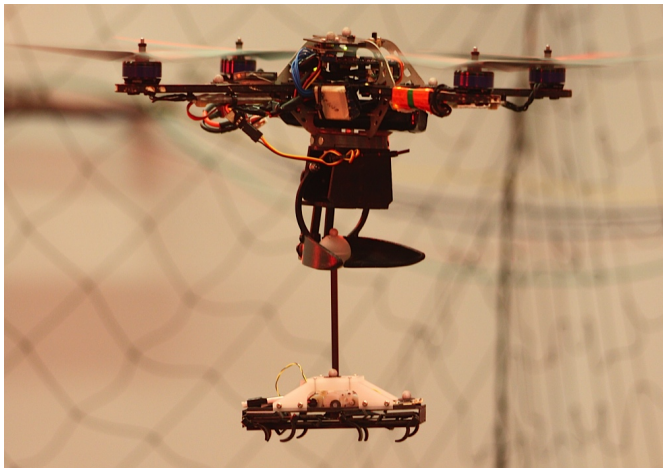

Fig. 1. Quadrotor, Gripper, and Grasped Object

$$F_i = k_f \omega_i^2 \tag{1}$$

$$M_i = \pm k_m \omega_i^2 \tag{2}$$

where the sign of k_m is based on the direction of rotation of the rotor. These relationships are used in [6] and [9], which are consistent with [10] for hover and vertical flight conditions.

The control inputs of the system are

$$\mathbf{u} = \begin{bmatrix} u_1 & u_2 & u_3 & u_4 \end{bmatrix}^T \in \mathbb{R}^4$$

and the rotor speeds are related to the control inputs through the following relationship

$$\begin{bmatrix} u_1 \\ u_2 \\ u_3 \\ u_4 \end{bmatrix} = \begin{bmatrix} k_f & k_f & k_f & k_f \\ 0 & lk_f & 0 & -lk_f \\ -lk_f & 0 & lk_f & 0 \\ k_m & -k_m & k_m & -k_m \end{bmatrix} \begin{bmatrix} \omega_1^2 \\ \omega_2^2 \\ \omega_3^2 \\ \omega_4^2 \end{bmatrix}. \tag{3}$$

An inertia tensor, $J \in \mathbb{R}^{3\times3}$, and mass, $m \in \mathbb{R}$, are used to model the robot's dynamics. Defining $\mathbf{r} \in \mathbb{R}^3$ as the position in $\mathcal{A}$, gravity by $g \in \mathbb{R}$, and $\mathbf{e}_3 = \begin{bmatrix} 0 & 0 & 1 \end{bmatrix}^T$, the translational dynamics are given by

$$m\ddot{\mathbf{r}} = u_1 R \mathbf{e}_3 - mg\mathbf{a}_3 \tag{4}$$

where $R \in SO(3)$ and rotates vectors from the body frame to the world frame.

The angular velocity vector of the body frame, $\mathcal{B}$, in the world frame, $\mathcal{A}$, expressed in the body frame coordinates is ${}^{\mathcal{B}}\mathbf{\Omega}^{\mathcal{B}} \in \mathbb{R}^3$ where the post-superscript indicates the frame of interest and the preceding superscript indicates the frame

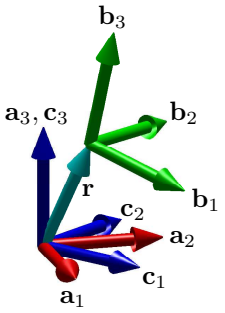

Fig. 2. Coordinate Systems

in which the coordinates are expressed. Then, the angular dynamics are

$$J\,{}^{\mathcal{B}}\dot{\mathbf{\Omega}}^{\mathcal{B}} = \begin{bmatrix} u_2 \\ u_3 \\ u_4 \end{bmatrix} - {}^{\mathcal{B}}\mathbf{\Omega}^{\mathcal{B}} \times J\,{}^{\mathcal{B}}\mathbf{\Omega}^{\mathcal{B}}. \tag{5}$$

We will define the state vector, $\mathbf{q} \in \mathbb{R}^{12}$, as

$$\mathbf{q} = \begin{bmatrix} \mathbf{r} \\ \dot{\mathbf{r}} \\ \boldsymbol{\gamma} \\ {}^{\mathcal{B}}\mathbf{\Omega}^{\mathcal{B}} \end{bmatrix} \tag{6}$$

where $\boldsymbol{\gamma} = \begin{bmatrix} \phi & \theta & \psi \end{bmatrix}^T \in \mathbb{R}^3$ and the components are roll, pitch, and yaw, respectively.

B. Differential Flatness

From the previous section, we have seen that the state space of a quadrotor is 12 dimensional consisting of the three components of $\mathbf{r}$ and their first derivatives, as well as the three components of angular orientation and their derivatives. Planning in this state space is a daunting task and so we seek a simpler space.

Initiated by [11] and further developed in [12], the community has utilized differential flatness to simplify the configuration space of the quadrotor. This concept has been specifically promoted in [13] and [9].

A system is differentially flat if there exists a change of coordinates which allows the state, $\mathbf{q}$, and control inputs, $\mathbf{u}$, to be written as functions of the flat outputs and their derivatives $(y_i, \dot{y}_i, \ddot{y}_i, ...)$ [12]. Additionally, we require that the flat outputs are functions of the state and the control inputs [12]. If the change of coordinates is a diffeomorphism, we can plan using the flat outputs and their derivatives.

1) Change of Coordinates: This section will show that the flat outputs of the quadrotor consist of $\mathbf{r}$ and the angular rotation, $\psi \in \mathbb{R}$, about $\mathbf{a}_3$. The flat outputs, $\mathbf{y} \in \mathbb{R}^4$, are defined

$$\mathbf{y} = \begin{bmatrix} \mathbf{r} \\ \psi \end{bmatrix}. \tag{7}$$

A simpler approach, which assumed instantaneous attitude control, was explained in [13]. Our approach, however, is similar to [9] and assumes that the rotors can instantaneously change their speed.

First, $\mathbf{r}$ and $\dot{\mathbf{r}}$ of the state vector are trivially populated with $y_1, y_2, y_3, \dot{y}_1, \dot{y}_2$, and $\dot{y}_3$. This leaves $\boldsymbol{\gamma}$ and ${}^{\mathcal{B}}\mathbf{\Omega}^{\mathcal{B}}$ as unknowns.

Solving (4) for $R\mathbf{e}_3$, the body frame axis $\mathbf{b}_3$ in $\mathcal{A}$, we obtain

$$\frac{m}{u_1}\left(\ddot{\mathbf{r}} + g\mathbf{a}_3\right) = R\mathbf{e}_3 \tag{8}$$

where $R\mathbf{e}_3 = \mathbf{b}_3$.

Since u_1 and m are scalars that do not affect the direction of $\mathbf{b}_3$, we simply have

$$\mathbf{b}_3 = \frac{\ddot{\mathbf{r}} + g\mathbf{a}_3}{\|\ddot{\mathbf{r}} + g\mathbf{a}_3\|} \tag{9}$$

and

$$u_1 = m\,\|\ddot{\mathbf{r}} + g\mathbf{a}_3\| \tag{10}$$

where we require that $\|\ddot{\mathbf{r}} + g\mathbf{a}_3\| > 0$.

Next, we wish to determine the other body frame axes. Since we know the yaw, y_4, we can define an intermediate coordinate frame, $\mathcal{C}$, with the vectors

$$\mathbf{c}_1 = \begin{bmatrix} \cos y_4 \\ \sin y_4 \\ 0 \end{bmatrix}, \quad \mathbf{c}_2 = \begin{bmatrix} -\sin y_4 \\ \cos y_4 \\ 0 \end{bmatrix}, \quad \mathbf{c}_3 = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}.$$

Then, $\mathbf{c}_2 \times \mathbf{b}_3$ generates a vector which is in the same direction as $\mathbf{b}_1$. Assuming that $\mathbf{b}_3 \neq \mathbf{c}_2$,

$$\mathbf{b}_1 = \frac{\mathbf{c}_2 \times \mathbf{b}_3}{\|\mathbf{c}_2 \times \mathbf{b}_3\|} \tag{11}$$

Finally, $\mathbf{b}_2$ is simply

$$\mathbf{b}_2 = \mathbf{b}_3 \times \mathbf{b}_1. \tag{12}$$

Noticing that $R \in SO(3)$ is simply a change of coordinates, we can write

$$R = \begin{bmatrix} \mathbf{b}_1 & \mathbf{b}_2 & \mathbf{b}_3 \end{bmatrix} \tag{13}$$

from which we can determine ϕ and θ, or q_7, and q_8 using the fact that R can also be defined using a Z-Y-X rotation

$$R = \begin{bmatrix} c\theta c\psi & c\psi s\theta s\phi - c\phi s\psi & c\phi c\psi s\theta + s\phi s\psi \\ c\theta s\psi & c\phi c\psi + s\theta s\phi s\psi & -c\psi s\phi + c\phi s\theta s\psi \\ -s\theta & c\theta s\phi & c\theta c\phi \end{bmatrix} \tag{14}$$

where s and c denote sin and cos, respectively.

Next, we seek ${}^{\mathcal{B}}\mathbf{\Omega}^{\mathcal{B}}$. Taking the first derivative of (4) with respect to time, we obtain

$$m\dddot{\mathbf{r}} = \dot{u}_1\mathbf{b}_3 + {}^{\mathcal{A}}\mathbf{\Omega}^{\mathcal{B}} \times u_1\mathbf{b}_3 \tag{15}$$

where ${}^{\mathcal{A}}\mathbf{\Omega}^{\mathcal{B}}$ is the angular velocity of the body in the inertial frame coordinates. Noticing that $\dot{u}_1$ is only multiplied by $\mathbf{b}_3$, we can solve for $\dot{u}_1$ by projecting (15) onto $\mathbf{b}_3$

$$\mathbf{b}_3 \cdot m\dddot{\mathbf{r}} = \mathbf{b}_3 \cdot \dot{u}_1\mathbf{b}_3 + \mathbf{b}_3 \cdot {}^{\mathcal{A}}\mathbf{\Omega}^{\mathcal{B}} \times u_1\mathbf{b}_3.$$

Since $\mathbf{b}_3 \perp \left({}^{\mathcal{A}}\mathbf{\Omega}^{\mathcal{B}} \times u_1\mathbf{b}_3\right)$, the last term vanishes giving

$$\dot{u}_1 = \mathbf{b}_3 \cdot m\dddot{\mathbf{r}}. \tag{16}$$

Substituting (16) into (15),

$$m\dddot{\mathbf{r}} = \left(\mathbf{b}_3 \cdot m\dddot{\mathbf{r}}\right)\mathbf{b}_3 + {}^{\mathcal{A}}\mathbf{\Omega}^{\mathcal{B}} \times u_1\mathbf{b}_3$$

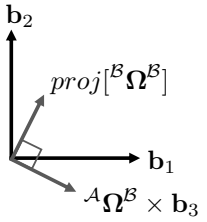

Fig. 3. Geometry of the relationship between $proj\left[{}^{\mathcal{B}}\mathbf{\Omega}^{\mathcal{B}}\right]$ and ${}^{\mathcal{A}}\mathbf{\Omega}^{\mathcal{B}} \times \mathbf{b}_3$ in the $\mathbf{b}_1$-$\mathbf{b}_2$ plane

we notice that $(\mathbf{b}_3 \cdot m\ddot{\mathbf{r}})\,\mathbf{b}_3$ is simply the component of $m\ddot{\mathbf{r}}$ along $\mathbf{b}_3$. Simplifying,

$$ {}^{\mathcal{A}}\mathbf{\Omega}^{\mathcal{B}} \times \mathbf{b}_3 = \frac{m}{u_1}\left(\dddot{\mathbf{r}} - (\mathbf{b}_3 \cdot \ddot{\mathbf{r}})\,\mathbf{b}_3\right) \tag{17} $$

so that $(\dddot{\mathbf{r}} - (\mathbf{b}_3 \cdot \ddot{\mathbf{r}})\,\mathbf{b}_3)$ and ${}^{\mathcal{A}}\mathbf{\Omega}^{\mathcal{B}} \times \mathbf{b}_3$ lie in the $\mathbf{b}_1$-$\mathbf{b}_2$ plane.

Since

$$ \left|{}^{\mathcal{A}}\mathbf{\Omega}^{\mathcal{B}} \times \mathbf{b}_3\right| = \left|{}^{\mathcal{A}}\mathbf{\Omega}^{\mathcal{B}}\right| |\mathbf{b}_3| \sin\eta = \left|{}^{\mathcal{A}}\mathbf{\Omega}^{\mathcal{B}}\right| \sin\eta $$

where η is the angle between ${}^{\mathcal{A}}\mathbf{\Omega}^{\mathcal{B}}$ and $\mathbf{b}_3$, we see that

$$ \left|{}^{\mathcal{A}}\mathbf{\Omega}^{\mathcal{B}} \times \mathbf{b}_3\right| = \left|proj\left[{}^{\mathcal{B}}\mathbf{\Omega}^{\mathcal{B}}\right]\right| $$

where $proj\left[{}^{\mathcal{B}}\mathbf{\Omega}^{\mathcal{B}}\right]$ is ${}^{\mathcal{B}}\mathbf{\Omega}^{\mathcal{B}}$ projected onto the $\mathbf{b}_1$-$\mathbf{b}_2$ plane. Additionally, the vectors are perpendicular as a result of the cross product. From Figure 3, we see that

$$ {}^{\mathcal{B}}\Omega_1^{\mathcal{B}} = -\mathbf{b}_2 \cdot \left(\frac{m}{u_1}\left(\dddot{\mathbf{r}} - (\mathbf{b}_3 \cdot \ddot{\mathbf{r}})\,\mathbf{b}_3\right)\right) \tag{18} $$

$$ {}^{\mathcal{B}}\Omega_2^{\mathcal{B}} = \mathbf{b}_1 \cdot \left(\frac{m}{u_1}\left(\dddot{\mathbf{r}} - (\mathbf{b}_3 \cdot \ddot{\mathbf{r}})\,\mathbf{b}_3\right)\right). \tag{19} $$

The last component of ${}^{\mathcal{B}}\mathbf{\Omega}^{\mathcal{B}}$ can be found using the relationship that the body frame components of angular velocity can be expressed in terms of Euler angles. Using (14), we can determine the body frame angular velocity according to [14] to be

$$ {}^{\mathcal{B}}\mathbf{\Omega}^{\mathcal{B}} = \left(R^T(t)\dot{R}(t)\right)^{\vee} $$

where $\cdot^{\vee} : \mathfrak{so}(3) \to \mathbb{R}^3$. This provides

$$ {}^{\mathcal{B}}\mathbf{\Omega}^{\mathcal{B}} = \begin{bmatrix} 1 & 0 & -\sin\theta \\ 0 & \cos\phi & \cos\theta\sin\phi \\ 0 & -\sin\phi & \cos\theta\cos\phi \end{bmatrix} \begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix} \tag{20} $$

Since ψ and $\dot{\psi}$ are given by y_4 and $\dot{y}_4$, respectively, and we know ϕ, θ, ${}^{\mathcal{B}}\Omega_1^{\mathcal{B}}$, and ${}^{\mathcal{B}}\Omega_2^{\mathcal{B}}$, we can solve for ${}^{\mathcal{B}}\Omega_3^{\mathcal{B}}$

$$ {}^{\mathcal{B}}\Omega_3^{\mathcal{B}} = \dot{\psi}\cos\theta\sec\phi - {}^{\mathcal{B}}\Omega_2^{\mathcal{B}}\tan\phi. \tag{21} $$

Lastly, we will determine u_2, u_3, and u_4. Taking the time derivative of (15),

$$ \begin{aligned} m\mathbf{r}^{(4)} = \ddot{u}_1\mathbf{b}_3 + 2\,{}^{\mathcal{A}}\mathbf{\Omega}^{\mathcal{B}} \times \dot{u}_1\mathbf{b}_3 + {}^{\mathcal{A}}\dot{\mathbf{\Omega}}^{\mathcal{B}} \times u_1\mathbf{b}_3 \\ + {}^{\mathcal{A}}\mathbf{\Omega}^{\mathcal{B}} \times {}^{\mathcal{A}}\mathbf{\Omega}^{\mathcal{B}} \times u_1\mathbf{b}_3 \end{aligned} \tag{22} $$

and projecting onto $\mathbf{b}_3$,

$$ \ddot{u}_1 = \mathbf{b}_3 \cdot m\mathbf{r}^{(4)} - \mathbf{b}_3 \cdot \left({}^{\mathcal{A}}\mathbf{\Omega}^{\mathcal{B}} \times {}^{\mathcal{A}}\mathbf{\Omega}^{\mathcal{B}} \times u_1\mathbf{b}_3\right). \tag{23} $$

Using $\ddot{u}_1$, (22) can be solved first for ${}^{\mathcal{A}}\dot{\mathbf{\Omega}}^{\mathcal{B}}$ and then ${}^{\mathcal{B}}\dot{\mathbf{\Omega}}^{\mathcal{B}}$ in a manner very similar to the determination of ${}^{\mathcal{B}}\mathbf{\Omega}^{\mathcal{B}}$. Once we have ${}^{\mathcal{B}}\dot{\mathbf{\Omega}}^{\mathcal{B}}$, we can solve (5) for the inputs. These results will be dependent on $\ddot{\psi}$ since we will need the time derivative of (21). The inputs are

$$ \begin{bmatrix} u_2 \\ u_3 \\ u_4 \end{bmatrix} = J\,{}^{\mathcal{B}}\dot{\mathbf{\Omega}}^{\mathcal{B}} + {}^{\mathcal{B}}\mathbf{\Omega}^{\mathcal{B}} \times J\,{}^{\mathcal{B}}\mathbf{\Omega}^{\mathcal{B}}. \tag{24} $$

Thus, the entire state vector, $\mathbf{q}$, and control inputs, $\mathbf{u}$, can be determined using only the flat outputs and their derivatives. Further, the inverse is true. Namely, $\mathbf{y}$ and the first four derivatives can be determined from the state vector and the control inputs.

An important observation is that the mapping between the full state space of the quadrotor and the flat space is a diffeomorphism under the assumptions $\mathbf{b}_3 \neq \mathbf{c}_2$ and $u_1 > 0$. In other words, the mapping between the flat space and the full state space is bijective and differentiable. Note that $u_1 \leq 0$ corresponds to zero or negative thrust, which is either not controllable or physically impossible. This would correspond to a scenario such as free fall or larger than gravity acceleration downwards.

2) Optimality: From the previous section, we see that the 4^{th} derivative of $\mathbf{r}$ and the 2^{nd} derivative of ψ are required to determine the control inputs, which agrees with [9]. As a result, any four times differentiable trajectory on position and two times differentiable trajectory on yaw is one that is dynamically feasible. Coming back to the concept of differential flatness, we see that for $i = 1, 2, 3$ we can write

$$ \boldsymbol{\xi}_i = \begin{bmatrix} y_i \\ \dot{y}_i \\ \ddot{y}_i \\ \dddot{y}_i \end{bmatrix}. \tag{25} $$

This allows us to define the flat system as

$$ \dot{\boldsymbol{\xi}}_i = \begin{bmatrix} 0_{3\times1} & I_{3\times3} \\ 0_{1\times1} & 0_{1\times3} \end{bmatrix} \boldsymbol{\xi}_i + \begin{bmatrix} 0_{3\times1} \\ 1 \end{bmatrix} y_i^{(4)}, \tag{26} $$

which is simply a chain of integrators with input $y_i^{(4)}$, or the snap of the component. Next, $\boldsymbol{\xi}_4$ depends only on two derivatives giving

$$ \boldsymbol{\xi}_4 = \begin{bmatrix} y_4 \\ \dot{y}_4 \end{bmatrix} \tag{27} $$

and

$$ \dot{\boldsymbol{\xi}}_4 = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} \boldsymbol{\xi}_4 + \begin{bmatrix} 0 \\ 1 \end{bmatrix} y_4^{(2)}. \tag{28} $$

Since the inputs are dependent on $y_i^{(4)}$ for $i = 1, 2, 3$ and $y_i^{(2)}$ for $i = 4$, we see that minimizing the snap of position and the acceleration of yaw over the trajectory will minimize the required control inputs. This is the same result developed in [9].

III. Trajectory Generation

Now that the system has been simplified, we can begin to plan in the flat space. The main inspiration for this approach comes from [9]. It is desirable that trajectories can be planned quickly and yet have flexibility. For example, straight line trajectories are possible, but they certainly would not be ideal in a maze of obstacles and waypoints. Throughout this section, it will be useful to define a trajectory using the basis

$$\mathbf{g}^T = \begin{bmatrix} 1 & t & t^2 & \dots & t^n \end{bmatrix}$$

with coefficients

$$\mathbf{c}^T = \begin{bmatrix} \alpha_0 & \alpha_1 & \alpha_2 & \dots & \alpha_n \end{bmatrix}.$$

Thus, the trajectory for one component is $\mathbf{g}^T\mathbf{c}$. Note that $\mathbf{c}$ is not related to $\mathbf{c}_i$ from the last section.

The challenge of planning trajectories that minimize the 4^{th} derivative of position and the 2^{nd} derivative of yaw over the duration of the trajectory motivates the cost functionals

$$J_i = \int_{t_0}^{t_f} \left\| y_i^{(4)}(t) \right\|^2 dt \quad \text{for} \quad i = 1, 2, 3 \tag{29}$$

and

$$J_4 = \int_{t_0}^{t_f} \left\| y_4^{(2)}(t) \right\|^2 dt, \tag{30}$$

which are intentionally quadratic. Note that J_i is not the same as the inertia tensor, J, used previously.

The trajectory generation methods presented in this paper are based on polynomial trajectories, but are not restricted to these alone. For example, trigonometric functions such as $\sin$ and $\cos$ can be used since they are C^∞ differentiable. The following sections are developed for the minimum snap components of the flat outputs. The minimum acceleration component can be derived similarly.

A. Euler-Lagrange

The most basic approach to optimize the trajectory is using the Euler-Lagrange equations:

$$\frac{\partial \mathcal{L}}{\partial f} + \sum_{k=1}^{n} (-1)^k \frac{d^k}{dt^k} \frac{\partial \mathcal{L}}{\partial f^{(k)}} = 0. \tag{31}$$

This relationship gives some insight into how we can plan trajectories which minimize the cost functionals (29) and (30). For the first three flat outputs, we have

$$f = y_i \quad \text{and} \quad \mathcal{L} = \left(y_i^{(4)} \right)^2$$

yielding

$$y_i^{(8)} = 0, \tag{32}$$

which reveals that a necessary condition for minimum snap is (32). A similar result for $i = 4$ shows that the 4^{th} derivative must be equal to 0 for yaw. In practice, using a 7^{th} order polynomial for position and a 3^{rd} order polynomial for yaw will satisfy these necessary conditions and generates the optimal trajectory given the appropriate boundary conditions.

Generating the coefficients, $\mathbf{c}$, can be done quite efficiently with the inversion of a single matrix of basis vectors:

$$\begin{bmatrix} \mathbf{g}(t_0)^T \\ \mathbf{g}^{(1)}(t_0)^T \\ \mathbf{g}^{(2)}(t_0)^T \\ \mathbf{g}^{(3)}(t_0)^T \\ \mathbf{g}(t_f)^T \\ \mathbf{g}^{(1)}(t_f)^T \\ \mathbf{g}^{(2)}(t_f)^T \\ \mathbf{g}^{(3)}(t_f)^T \end{bmatrix} \mathbf{c} = \begin{bmatrix} y_i(t_0) \\ y_i^{(1)}(t_0) \\ y_i^{(2)}(t_0) \\ y_i^{(3)}(t_0) \\ y_i(t_f) \\ y_i^{(1)}(t_f) \\ y_i^{(2)}(t_f) \\ y_i^{(3)}(t_f) \end{bmatrix} \tag{33}$$

B. Quadratic Programming

A drawback from the previous approach is that the trajectory is strictly a 7^{th} order polynomial. Unfortunately, only boundary conditions can be specified, not intermediate waypoints since we require that the position and the first three derivatives are equal to the adjacent segment at the boundaries. For a single segment, the derivatives must be equal to 0. We require this continuity since the dynamics demand smooth derivatives below snap. For more flexibility, we seek higher order polynomials through the use of using quadratic programming.

We can explicitly write the integrands of the cost functionals

$$\mathbf{c}^T \frac{d^4\mathbf{g}}{dt^4} \left[\frac{d^4\mathbf{g}}{dt^4} \right]^T \mathbf{c}$$

and define

$$G = \int_{t_0}^{t_f} \frac{d\mathbf{g}^4}{dt^4} \left[\frac{d\mathbf{g}^4}{dt^4} \right]^T dt.$$

The problem is now posed as a quadratic optimization problem, or QP, where we want to minimize $\mathbf{c}^T G \mathbf{c}$ over the duration of the trajectory subject to some specified constraints.

1) Analytic Solution: If only equality constraints are present, this problem can be solved analytically. Consider:

$$\arg\min_{\mathbf{c}} \quad \mathbf{c}^T G \mathbf{c} \tag{34}$$

subject to

$$H\mathbf{c} = \mathbf{d} \tag{35}$$

It is worth noting that $G = G^T$, $\det(G) = 0$, and $G \succeq 0$. Incorporating the equality constraint using a vector of Lagrange multipliers, $\boldsymbol{\lambda}$, we obtain

$$\mathcal{L} = \mathbf{c}^T G \mathbf{c} + \boldsymbol{\lambda}^T (H\mathbf{c} - \mathbf{d}). \tag{36}$$

Then,

$$\frac{\partial \mathcal{L}}{\partial \mathbf{c}} = \mathbf{c}^T (G + G^T) + \boldsymbol{\lambda}^T H = 2\mathbf{c}^T G + \boldsymbol{\lambda}^T H.$$

Because the system is quadratic, we know that at the minimum

$$\frac{\partial \mathcal{L}}{\partial \mathbf{c}} = \mathbf{0}.$$

This provides another equation that we can use in conjunction with the equality constraint to solve for $\mathbf{c}$ and $\boldsymbol{\lambda}$. Without any loss, we can transpose the equation to obtain

$$2G\mathbf{c} + H^T\boldsymbol{\lambda} = \mathbf{0}$$

Finally, consider the following set of linear equations:

$$\begin{bmatrix} 2G & H^T \\ H & \mathbf{0} \end{bmatrix} \begin{bmatrix} \mathbf{c} \\ \boldsymbol{\lambda} \end{bmatrix} = \begin{bmatrix} \mathbf{0} \\ \mathbf{d} \end{bmatrix}. \tag{37}$$

Then, the inverse of

$$\begin{bmatrix} 2G & H^T \\ H & \mathbf{0} \end{bmatrix}$$

can be used to estimate $\mathbf{c}$.

This approach enables us to minimize the snap (or acceleration) of trajectories higher than order 7 (or 3). It may require an approximation of the inverse of a singular matrix, but in practice, it works well.

2) Numerical Solution: Using H with the same equality constraints and inequality constraints in a similar form, the system can be solved numerically using modern quadratic programming solvers. Non-linear constraints such as thrust limitations or a circular bounding region can be implemented as a number of inequalities in the flat space.

C. Other Trajectory Generation Methods

Certainly the proposed trajectory generation methods are not the only ones. [9] proposes a solution that allows a trajectory to be scaled, however, in its current form, the velocities are also scaled, which would be undesirable for grasping and perching where velocities are important.

The generated trajectories are guaranteed to be dynamically feasible by the quadrotor, however, they do not consider actuator constraints. For example, the quadrotor can actuate to fly a trajectory that is four times differentiable, but it may not be feasible given the duration. Although there are several methods to deal with this problem, one will be discussed briefly.

A fairly recent concept is motion planning using Linear Quadratic Regulator (LQR) trees [15]. This approach builds upon Rapidly-exploring Random Trees (RRTs) in that random points in the state space are chosen and a tree is constructed towards the random point [16]. A major departure from RRTs, however, is that LQR trees consider the system dynamics and control inputs through the use of LQR controllers and a system that is linearized about the trajectory. A Lyapunov function is developed based on the LQR design from which a basin of attraction is determined. This is essentially a funnel in the state space that guarantees that the output (of the funnel) can be reached from any starting point within the funnel. Continuing to choose random points in the state space can probabilistically cover a desired region. It is interesting to note that LQR trees could be used to expand pre-existing optimal trajectories.

IV. Controllers

Now that a desired trajectory has been generated, we seek to control the robot to follow the trajectory. A traditional approach is to linearize the dynamics of the quadrotor about a given state and use either a PID or an LQR controller. Both of these will be presented below. Additionally, a non-linear controller on $SE(3)$ will be proposed. Each of these controllers have two key parts, an attitude controller and a position controller.

The PID and LQR controllers use the same simplification of the system dynamics. We generally linearize the system about hover, although other approaches such as linearization along the trajectory are possible.

Additionally, while a Feedback Linearization paradigm can be successful in simulation, it is not practical on real systems since it requires higher derivatives of position, which are typically very noisy.

A. PID Controller

A PID controller uses Proportional, Integral, and Derivative feedback on the error from the desired setpoints. This controller has been explained in detail in [2], [6], and has been an established method of control for many years. A cascaded controller is used with the attitude control as the slave and position control as the master as pictured in Figure 4.

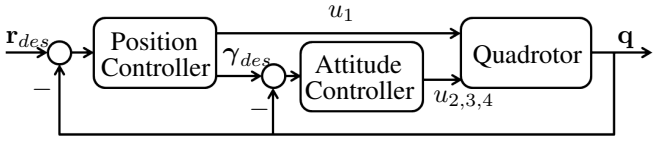

Fig. 4. PID Controller

1) Position Control: Since we are primarily concerned about the position control, we will specify an attitude that will drive toward the desired positions. The system dynamics from (4) are linearized about $\phi = 0$, $\theta = 0$, $\dot{\mathbf{r}} = \mathbf{0}$, and ${}^{\mathcal{B}}\dot{\boldsymbol{\Omega}}^{\mathcal{B}} = \mathbf{0}$ to obtain

$$\ddot{\mathbf{r}} = \frac{1}{m}\begin{bmatrix} u_1\left(\cos(\psi)\theta + \sin(\psi)\phi\right) \\ u_1\left(\sin(\psi)\theta - \cos(\psi)\phi\right) \\ u_1 - mg \end{bmatrix}$$

and noting that at hover, $u_1 \approx mg$, we can simplify the first two terms to get

$$\ddot{\mathbf{r}} = g\begin{bmatrix} \sin(\psi) & \cos(\psi) & 0 \\ -\cos(\psi) & \sin(\psi) & 0 \\ 0 & 0 & \frac{1}{mg} \end{bmatrix}\begin{bmatrix} \phi \\ \theta \\ u_1 \end{bmatrix} - g\mathbf{a}_3. \tag{38}$$

Defining errors on positions and velocities, the desired accelerations are

$$\begin{aligned} \ddot{r}_{i,des} = k_{p,i}^{pos}(y_{i,des} - r_i) + k_{i,i}^{pos}\int_{t_0}^{t_f}(y_{i,des} - r_i)dt \\ + k_{d,i}^{pos}(\dot{y}_{i,des} - \dot{r}_i) \end{aligned} \tag{39}$$

for $i = 1, 2, 3$ where $k_{p,i}^{pos}$, $k_{i,i}^{pos}$, and $k_{d,i}^{pos}$ are the gains on the position, integral, and derivative errors. Then, we can

determine $u_{1,des}$ and the first two terms of $\boldsymbol{\gamma}_{des}$ from (38). Solving for $\begin{bmatrix} \phi & \theta & u_1 \end{bmatrix}^T$,

$$\begin{bmatrix} \phi \\ \theta \\ u_1 \end{bmatrix} = \begin{bmatrix} \sin(\psi) & -\cos(\psi) & 0 \\ \cos(\psi) & \sin(\psi) & 0 \\ 0 & 0 & mg \end{bmatrix} \left(\frac{\ddot{\mathbf{r}}}{g} + \mathbf{a}_3 \right). \quad (40)$$

Then, the desired attitude from the position controller is

$$\gamma_{1,des} = \frac{1}{g} \left(\ddot{r}_{1,des} \sin(\psi) - \ddot{r}_{2,des} \cos(\psi) \right) \quad (41)$$

$$\gamma_{2,des} = \frac{1}{g} \left(\ddot{r}_{1,des} \cos(\psi) + \ddot{r}_{2,des} \sin(\psi) \right) \quad (42)$$

$$\gamma_{3,des} = \psi. \quad (43)$$

Additionally, we see from (40)

$$u_1 = m\ddot{r}_{3,des} + mg. \quad (44)$$

2) Attitude Control: Now that a desired attitude has been established, we must try to achieve it. First, we make the assumption that J is diagonal, which is appropriate since the mass of the quadrotor is located symmetrically along the principal axes. Then, the rotational dynamics from (5) can be linearized to obtain

$$\begin{bmatrix} u_{2,ff} \\ u_{3,ff} \\ u_{4,ff} \end{bmatrix} = J \, {}^{\mathcal{B}}\dot{\boldsymbol{\Omega}}^{\mathcal{B}}. \quad (45)$$

This result gives feed-forward control on the desired angular acceleration. Then, a control law can be written in the form

$$\begin{aligned} u_{i+1} = k_{p,i}^{att}(\gamma_{i,des} - \gamma_i) + k_{i,i}^{att} \int_{t_0}^{t_f} (\gamma_{i,des} - \gamma_i) dt \\ + k_{d,i}^{att} \left({}^{\mathcal{B}}\Omega_{i,des}^{\mathcal{B}} - {}^{\mathcal{B}}\Omega_i^{\mathcal{B}} \right) + u_{i+1,ff} \end{aligned} \quad (46)$$

for $i = 1, 2, 3$ where ${}^{\mathcal{B}}\Omega_{i,des}^{\mathcal{B}}$ is determined from the flat outputs of the trajectory and $k_{p,i}^{att}$, $k_{i,i}^{att}$, and $k_{d,i}^{att}$ are the gains.

B. Linear Quadratic Regulator

To develop the LQR controller, we need the system to be in state space form. Starting with (38), we treat the $g\mathbf{a}_3$ term as if it is part of the u_1 term.

Next, the inverse of the relationship given in (20) relates the body frame angular velocities and Euler angles

$$\begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix} = \begin{bmatrix} 1 & \sin(\phi)\tan(\theta) & \cos(\phi)\tan(\theta) \\ 0 & \cos(\phi) & -\sin(\phi) \\ 0 & \sec(\theta)\sin(\phi) & \cos(\phi)\sec(\theta) \end{bmatrix} {}^{\mathcal{B}}\boldsymbol{\Omega}^{\mathcal{B}}.$$

and using small angle approximations,

$$\begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix} = \mathbf{I}_{3\times 3} \, {}^{\mathcal{B}}\boldsymbol{\Omega}^{\mathcal{B}}.$$

Then, the linear system is given by

$$\dot{\mathbf{q}} = A\mathbf{q} + B\mathbf{u} \quad (47)$$

where

$$A = \begin{bmatrix} \mathbf{0}_{3\times 3} & \mathbf{I}_{3\times 3} & \mathbf{0}_{3\times 1} & \mathbf{0}_{3\times 1} & \mathbf{0}_{3\times 1} & \mathbf{0}_{3\times 3} \\ \mathbf{0}_{1\times 3} & \mathbf{0}_{1\times 3} & gs(\psi) & gc(\psi) & 0 & \mathbf{0}_{1\times 3} \\ \mathbf{0}_{1\times 3} & \mathbf{0}_{1\times 3} & -gc(\psi) & gs(\psi) & 0 & \mathbf{0}_{1\times 3} \\ \mathbf{0}_{1\times 3} & \mathbf{0}_{1\times 3} & 0 & 0 & 0 & \mathbf{0}_{1\times 3} \\ \mathbf{0}_{3\times 3} & \mathbf{0}_{3\times 3} & 0 & 0 & 0 & \mathbf{I}_{3\times 3} \\ \mathbf{0}_{3\times 3} & \mathbf{0}_{3\times 3} & 0 & 0 & 0 & \mathbf{0}_{3\times 3} \end{bmatrix}$$

and

$$B = \begin{bmatrix} \mathbf{0}_{5\times 1} & \mathbf{0}_{5\times 1} & \mathbf{0}_{5\times 1} & \mathbf{0}_{5\times 1} \\ \frac{1}{m} & 0 & 0 & 0 \\ \mathbf{0}_{3\times 1} & \mathbf{0}_{3\times 1} & \mathbf{0}_{3\times 1} & \mathbf{0}_{3\times 1} \\ 0 & \frac{1}{J_{xx}} & 0 & 0 \\ 0 & 0 & \frac{1}{J_{yy}} & 0 \\ 0 & 0 & 0 & \frac{1}{J_{zz}} \end{bmatrix}$$

which is a controllable system. The control inputs can then be written in the form

$$\mathbf{u} = K(\mathbf{q}_{des} - \mathbf{q}) + \begin{bmatrix} mg \\ 0 \\ 0 \\ 0 \end{bmatrix} \quad (48)$$

where $K \in \mathbb{R}^{n\times n}$ is a gain matrix. The state vector is the output of the linear system so the transfer function becomes

$$G(s) = (sI - A)^{-1}B \quad (49)$$

and the system is observable. The optimal gain matrix, K, is determined by minimizing the cost functional

$$J = \int_{t_0}^{\infty} \left(\mathbf{q}^T Q \mathbf{q} + \mathbf{u}^T R \mathbf{u} \right) dt \quad (50)$$

where $Q, R \succeq 0$ [17]. In order for Q and R to be on similar scales, we use Bryson's Rule

$$R = diag\left[u_{1,max}^{-2},\ u_{2,max}^{-2},\ u_{3,max}^{-2},\ u_{4,max}^{-2} \right] \quad (51)$$

and a similar approach is used for Q. Then, K can be determined by solving the Algebraic Riccati Equation [17]

$$PA + A^T P + Q - PBR^{-1}B^T P = 0$$

where $K = R^{-1}B^T P$.

C. SE(3) Controller

The previous two controllers required linearization of the system dynamics. This is obviously not desirable and is also unnecessary. Using the same dynamic model presented earlier, [18] develops a non-linear controller on $SE(3)$ which is proven to have near-global exponential attractiveness using a Lyapunov analysis.

1) Position Controller: As before, the position controller will simply establish a desired attitude. First, we define $\mathbf{e}_\mathbf{r} = \mathbf{r}_{des} - \mathbf{r}$ and $\dot{\mathbf{e}}_\mathbf{r} = \dot{\mathbf{r}}_{des} - \dot{\mathbf{r}}$. Then we establish $\mathbf{b}_{3,des}$ as

$$\mathbf{b}_{3,des} = \frac{k_p \mathbf{e}_r + k_d \dot{\mathbf{e}}_r + mg\mathbf{a}_3 + m\ddot{\mathbf{r}}_d}{\| k_p \mathbf{e}_r + k_d \dot{\mathbf{e}}_r + mg\mathbf{a}_3 + m\ddot{\mathbf{r}}_d \|}. \quad (52)$$

This is nearly identical to [18] except that we are using $\mathbf{b}_3$ up instead of down. Since the desired yaw is y_4, we can determine

R_d as in II-B starting with (8). Then, the desired thrust is given by

$$u_1 = [k_p \mathbf{e}_r + k_d \dot{\mathbf{e}}_r + mg\mathbf{a}_3 + m\ddot{\mathbf{r}}_d] \cdot \mathbf{b}_3 \tag{53}$$

where the dot product with $\mathbf{b}_3$ ensures that the total thrust is inversely proportional to the attitude error. This gives priority to the attitude controller.

2) Attitude Controller: First, an error function is defined on $SO(3)$ as

$$\Psi(R, R_d) = \frac{1}{2} tr\left[I - R_d^T R\right] \tag{54}$$

where R_d rotates from the desired attitude to the world coordinates and R rotates from the body frame to the world coordinates. Since this may not be intuitive, the implications will be discussed. The tr operator is linear so that (54) becomes

$$\Psi(R, R_d) = \frac{1}{2}\left(tr\left[I\right] - tr\left[R_d^T R\right]\right) = \frac{1}{2}\left(3 - tr\left[R_d^T R\right]\right).$$

Using Rodrigues' Formula, the tr of $R_d^T R \in SO(3)$ is given by $1+2\cos\zeta$ where in axis-angle representation, ζ is the angle of rotation. Then, the functional looks like

$$\Psi(R, R_d) = 1 - \cos\zeta.$$

Now, we see that $\Psi = 0$ when $\zeta = 0$ and Ψ is maximized when $\zeta = \pm\pi$.

Next, the minimum of the functional can be found by noting that $R_d^T R$ is a function of time. Defining $R_d^T R$ as a rotation matrix whose time derivative is given by $R_d^T R\hat{\eta}$,

$$\frac{d\Psi}{dt} = \frac{1}{2} tr\left[\frac{d}{dt}\left(-R_d^T R\right)\right] = -\frac{1}{2} tr\left[R_d^T R\hat{\eta}\right], \tag{55}$$

which can be simplified to

$$\frac{d\Psi}{dt} = \frac{1}{2}\left(R_d^T R - R^T R_d\right)^\vee \cdot \eta. \tag{56}$$

where we have used the identity $tr(\hat{x}\hat{y}) = -2(x \cdot y)$. Then, it is understandable to define the error on the attitude as

$$\mathbf{e}_R = \frac{1}{2}\left(R_d^T R - R^T R_d\right)^\vee. \tag{57}$$

Next, we define ${}^{\mathcal{D}}\mathbf{\Omega}^{\mathcal{D}}$ as the desired angular velocity expressed in coordinates of the desired attitude frame, $\mathcal{D}$, so that the error on the angular velocity in the current body frame is

$$\mathbf{e}_\Omega = {}^{\mathcal{B}}\mathbf{\Omega}^{\mathcal{B}} - R^T R_d \, {}^{\mathcal{D}}\mathbf{\Omega}^{\mathcal{D}}. \tag{58}$$

Then, the attitude control is

$$\begin{aligned}\begin{bmatrix} u_2 \\ u_3 \\ u_4 \end{bmatrix} =& -k_R \mathbf{e}_R - k_\Omega \mathbf{e}_\Omega + {}^{\mathcal{B}}\mathbf{\Omega}^{\mathcal{B}} \times J \, {}^{\mathcal{B}}\mathbf{\Omega}^{\mathcal{B}} \\ & - J\left({}^{\mathcal{B}}\hat{\mathbf{\Omega}}^{\mathcal{B}} R^T R_d \, {}^{\mathcal{D}}\mathbf{\Omega}^{\mathcal{D}} - R^T R_d \, {}^{\mathcal{D}}\dot{\mathbf{\Omega}}^{\mathcal{D}}\right).\end{aligned} \tag{59}$$

The first two terms provide feedback control on the attitude and the angular velocity. The third term, ${}^{\mathcal{B}}\mathbf{\Omega}^{\mathcal{B}} \times J \, {}^{\mathcal{B}}\mathbf{\Omega}^{\mathcal{B}}$, is the current state, and the last term provides control on the angular acceleration.

This attitude controller is shown in [18] to be exponentially stable when

$$\Psi_0 < 2$$

and

$$\|\mathbf{e}_\Omega(0)\|^2 < \frac{2}{\lambda_{max}(J)} k_R \left(2 - \Psi_0\right). \tag{60}$$

Additionally, the tracking controller is shown to be exponentially stable when

$$\Psi_0 \leq 1 \tag{61}$$

and (60) are true. This corresponds to an initial geodesic angle error of less than $\pi/2$. Note that as the inertia or the initial angular velocity error increases, we require a higher gain, k_R.

A verbose version of the Lyapunov stability analysis for the attitude, translational, and total dynamics can be found in the appendix of [19]. Further, [14] is a comprehensive source of information pertaining to geometric control.

D. Summary of Controllers

While each controller has its benefits, such as the simplicity of a PID controller, optimality of LQR, and the non-linear control on SE(3), they do have downfalls. For example, the PID and LQR controllers require careful consideration of the Euler angles, particularly when the reference or actual angles approach branch cuts. While it is not difficult to implement corrective algorithms, the PID and LQR controllers still require a linearized system, which loses significant information about the system. On the other hand, the non-linear controller on SE(3) is challenging to prove, but does not suffer from singularities and has near global asymptotic attractiveness.

V. Other Aerodynamic Considerations

The dynamic model typically used in literature is based on the assumption that the rotors only generate forces and moments acting axially. While this is generally true for hovering and vertical flight, it is a well known fact in helicopter and rotor theory that this is not the case in forward flight [10]. Other aerodynamic factors include the angle of attack, which requires higher thrust to maintain altitude, differing relative velocities of the advancing and retreating blades, which creates blade flapping, and the disrupted airflow around the body of the quadrotor, which creates drag [20]. In light of such knowledge, [21] develops a controller that factors in the result of airflow parallel to the plane of the rotor. The body drag is ignored since it is related quadratically to the velocity, and therefore, it is negligible at low velocities. The difference between the "standard model" and the "proposed model" is considered insignificant in practice. However, this analysis only considers the rotor drag for an outdoor system and it is not a comprehensive model of the aerodynamics. A more applicable model can be developed for feed-forward control if we have better state estimation. For example, the consideration of the free stream velocity of the airflow perpendicular to the rotor plane has been shown to significantly improve tracking performance during ascent [22].

VI. Perching

Using the planning, trajectory generation, and control techniques outlined in this paper, the perching problem, in its simplest form, becomes fairly straightforward. In general,

perching on horizontal surfaces will necessitate minimal translational velocities and only a small vertical velocity at the time of impact. Perching is then reduced to the problem of planning a trajectory from the current state to the perch point that terminates with non-zero boundary conditions. We can enforce further constraints that will alter the aggressiveness of the trajectory such as the desired acceleration at impact or the maximum snap along the trajectory. One important consideration is the rate of descent. Recall that to maintain control, we require $u_1 > 0$. While this is always a consideration, it will particularly manifest itself when minimizing perching time. Additionally, if u_1 is small, the position control would require greater angular deviations from hover, which is not desirable especially near a perch location. Thus, we require $u_1 > \epsilon > 0$ where ϵ is determined experimentally.

VII. Grasping

Grasping is highly dependent on the design of the gripper and therefore, one particular approach will be explained. The gripper used in our experiments was designed with the intention of grasping man-made objects with minimal effort. A sphere was attached to a rod, which was attached to the target. This design allows omni-directional approaches on the target. The quadrotor, gripper, and grasped object are shown in Figure 1.

Now we have added another segment to the trajectory since grasping does not end at the pickup location. A final position is specified and we must optimize both segments of the trajectory. It is worth noting that it would be straightforward to use a single polynomial and specify the target location as an intermediate equality constraint if a QP approach was used. Additionally, computing a single polynomial for the entire trajectory may decrease computation time.

As with perching, grasping requires certain constraints such as a specified velocity at the pickup location. For example, the grasping success rate of the gripper was significantly improved with a slight velocity in the positive $\mathbf{a}_1$ direction.

VIII. Results

A. Trajectory Generation

The relative durations of the three different trajectory generation methods have been summarized in Table I. Sample snap profiles are provided in Figure 5.

TABLE I
Duration of Trajectory Generation for 1 Iteration

	Calc. Time (ms)
Euler-Lagrange	127
QP Equality	115
Numerical QP	1549

Obviously, the numerical solution is not practical for quick planning. A custom solver, however, may decrease the computational time. It is somewhat unexpected that the analytic QP is faster than the Euler-Lagrange solution. This variation may be a result of differences in code optimization. Regardless, the first two methods are fast enough that they can be used in flight to generate trajectories.

The three methods return the same result for equivalent boundary conditions as seen in Figure 5. However, the flexibility of the quadratic programming approaches allows for specifying intermediate boundary conditions so that the trajectory generated in Figure 10 could be generated as a single trajectory instead of two separate trajectories pieced together as required with the Euler-Lagrange solution.

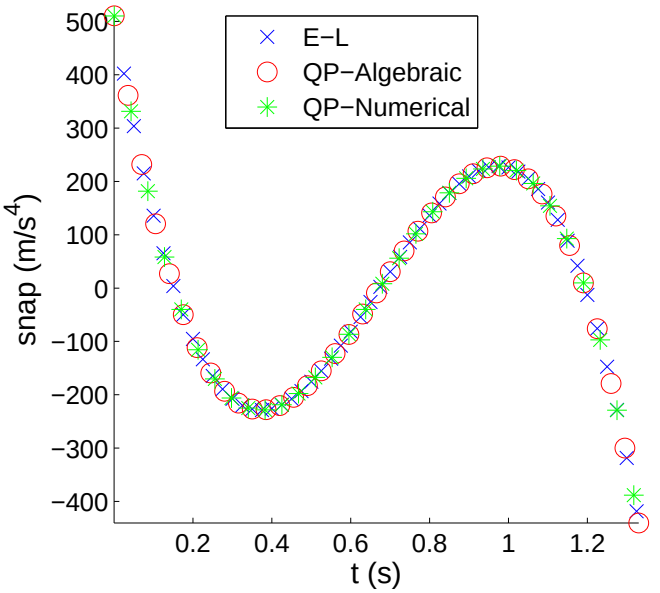

Fig. 5. The snap of the three trajectory generators is compared for one component of a sample trajectory. The boundary conditions are the same among generators and there is no time optimization. The Euler-Lagrange solution is a 7th order polynomial while the QP solutions are permitted to generate 11^{th} order polynomials. Note that the three solutions are identical.

B. Controllers

The same trajectory was flown in simulation using a PD controller in Figure 6 and the LQR controller in Figure 7. This particular trajectory has two segments that are joined at the pickup location. The boundary positions are specified, and the boundary velocities, accelerations, and jerks are zero. Lateral and vertical velocities are specified at the pickup location, and the angle of approach is computed to be the same angle of the line segment between the start and finish positions. To ensure that the quadrotor is level when grasping, the acceleration and jerk at the pickup location are required to be zero.

The PD and LQR controllers yield similar results. However, their performance is highly dependent on tuning. In general, the LQR controller is easier to tune because we are only adjusting one tuning parameter, a scaling factor between Q and R, instead of eight as with the PD controller.

C. Perching

The perching and trajectory generation capabilities were demonstrated in an experiment that allowed for random interaction from a user. The quadrotor was driven in a straight

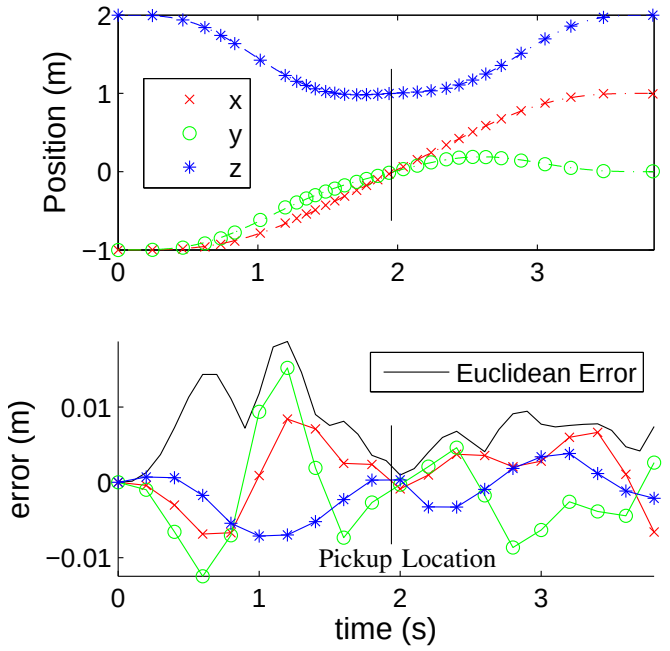

Fig. 6. Position tracking performance of the PD controller. The X, Y, and Z positions in the world frame (top) are optimized for the dynamics of the quadrotor. The error for each component as well as the Euclidean error are give on the bottom.

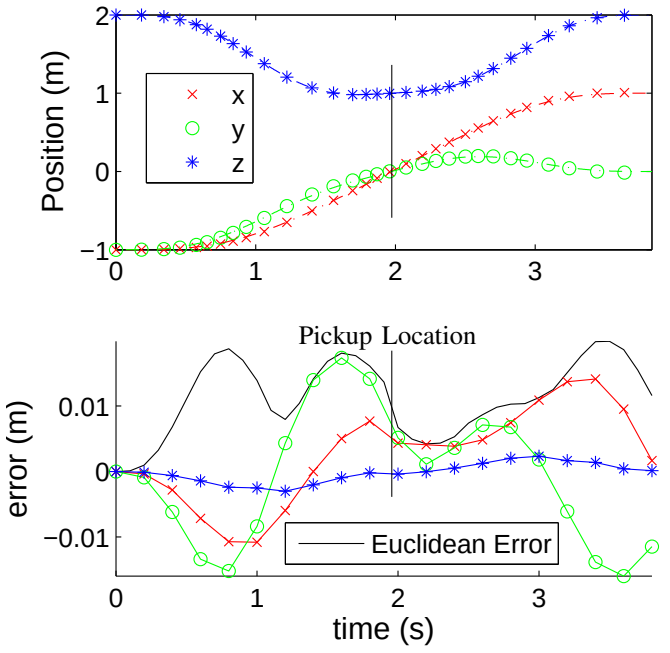

Fig. 7. Position tracking performance of the LQR controller for the same trajectory as Figure 6. The errors are similar in magnitude to the PD controller.

line towards an arbitrary point. Then, at a random time along the trajectory, the quadrotor was commanded to perch. Considering the initial conditions and the desired final conditions, a minimal snap trajectory was planned in-flight to drive the robot to the perch location. A sample trajectory projected onto the $X-Z$ axes has been provided in Figure 8 and the corresponding attitude is displayed in Figure 9. An expected result was observed; the error at the end of the trajectory is larger than the rest of the trajectory. This likely a result of ground effect, which causes an increase in the net thrust as the quadrotor approaches the ground [23], [22].

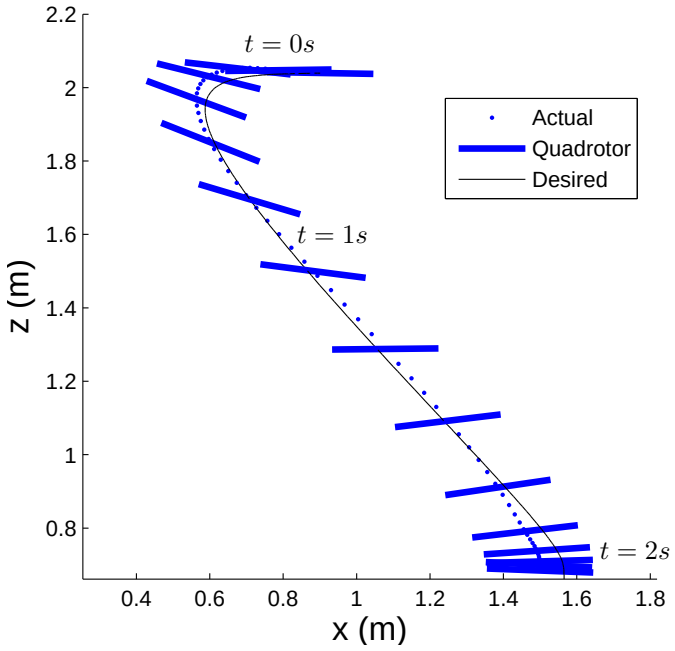

Fig. 8. The trajectory of a quadrotor perching using PID control after being interrupted during the straight line maneuver. This is a projection onto world X-Z plane. The perching trajectory considered the initial velocity in the $-X$ direction and specified a small final velocity in the $-Z$ direction.

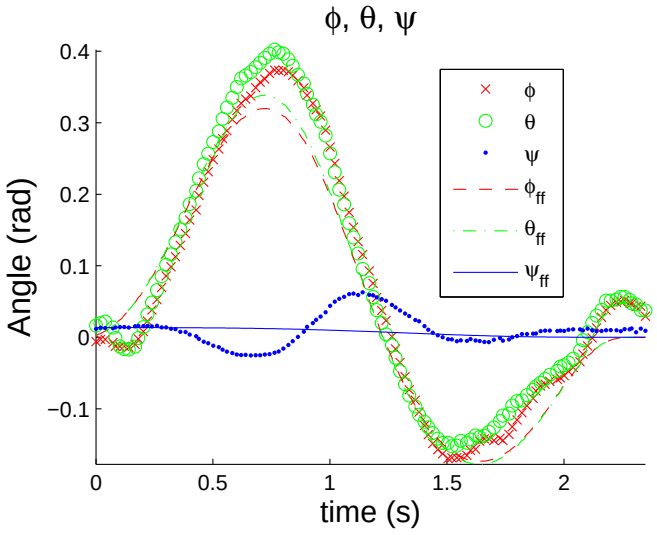

Fig. 9. Roll, Pitch, and Yaw of the perching trajectory in Figure 8. The feed-forward control is indicated by lines and the actual angles are indicated by markers.

D. Grasping

In addition to perching, this trajectory planning method allows for multi-segmented trajectories necessary for grasping.

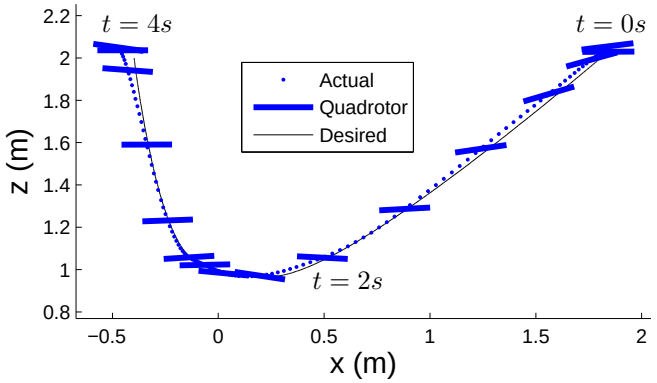

Fig. 10. A grasping trajectory projected onto the world X-Z plane. The quadrotor orientation is overlayed and scaled such that the *Quadrotor Lines* are the projected distance from the front rotor to the rear rotor. This experiment was conducted with a real quadrotor without picking up a target.

To demonstrate this, a target location in the workspace was given and a final position was chosen randomly. While flying in a random direction, the quadrotor was commanded to plan an optimal path to the target and end at the randomly selected final location. Experiment results are presented in Figure 10.

In experiments with grasping, the quadrotor's dynamics change after pickup, and minor instabilities resulting from the pendulum-like swinging of the target are observed. In future work, the gripper will be designed to minimize swinging of the target, which will allow for compensation by anticipating the change in dynamics after target acquisition.

IX. Conclusion

The success and application of three trajectory generation methods have been demonstrated in this paper. Overall, the analytic QP solution has the best balance of capability and speed. However, each solution has its benefits, namely the simplicity of the Euler - Lagrange approach and the flexibility of the Quadratic Programming methods. Next, several controllers were presented; PID, LQR, and SE(3). The PID and LQR controllers are appropriate for minimal excursions from hover, but for more aggressive maneuvers, it will be beneficial to use the non-linear SE(3) controller. Finally, the ground effect resulting from perching on flat surfaces was observed and the implications of a new rotor model were briefly discussed.

X. Future Work

There are many possible directions for improvement. First, the controller developed in [18] will be implemented on the physical system. Additionally, in-flight replanning for acquisition of moving targets will be explored. This may utilize concepts such as preplanned sets of dynamically feasible trajectories. In an effort to reduce the reliance upon VICON, visual servoing will be used for target acquisition, tracking, and perching. A collaboration has been established to develop new grippers which will allow for perching on a larger variety of surfaces and grasping of arbitrary objects. Finally, the aerodynamics of ground effect and rotor drag need to be considered to improve the precision of perching as well as tracking control of aggressive trajectories.

XI. Acknowledgements

The author would like to thank Vijay Kumar for his guidance and patience throughout the development of this paper. Additionally, the author is grateful for Matt Turpin's encouragement and many discussions.

References

[1] Quentin Lindsey, Daniel Mellinger, and Vijay Kumar. Construction of Cubic Structures with Quadrotor Teams. In *Proceedings of Robotics: Science and Systems*, Los Angeles, CA, USA, 2011.

[2] Daniel Mellinger, Nathan Michael, and Vijay Kumar. Trajectory Generation and Control for Precise Aggressive Maneuvers with Quadrotors. In *Proceedings of the International Symposium on Experimental Robotics*, 2010.

[3] Vijay Kumar and Nathan Michael. Opportunities and challenges with autonomous micro aerial vehicles. *Int. Symp. on Robotics Research*, pages 1–16, 2011.

[4] Daniel Mellinger, Quentin Lindsey, Michael Shomin, and Vijay Kumar. Design, modeling, estimation and control for aerial grasping and manipulation. In *Intelligent Robots and Systems (IROS), 2011 IEEE/RSJ International Conference on*, pages 2668–2673. IEEE, 2011.

[5] Daniel Mellinger, Michael Shomin, and Vijay Kumar. Control of Quadrotors for Robust Perching and Landing. In *Proceedings of the International Powered Lift Conference*, 2010.

[6] Nathan Michael, Daniel Mellinger, Quentin Lindsey, and Vijay Kumar. The GRASP multiple micro-UAV testbed. *Robotics & Automation Magazine, IEEE*, 17(3):56–65, 2010.

[7] Vicon Motion Capture System.

[8] Ascending Technologies GmbH.

[9] Daniel Mellinger and Vijay Kumar. Minimum snap trajectory generation and control for quadrotors. In *2011 IEEE International Conference on Robotics and Automation*, pages 2520–2525. IEEE, May 2011.

[10] Alfred Gessow and Garry Myers. *Aerodynamics of the Helicopter*. Frederick Ungar Publishing Co., 1978.

[11] Michel Fliess, Jean Levine, Philippe Martin, and Pierre Rouchon. Flatness and defect of non-linear systems: introductory theory and examples. *International Journal of Control*, 61(6):1327–1361, June 1995.

[12] R.M. Murray, M. Rathinam, and Willem Sluis. Differential flatness of mechanical control systems: A catalog of prototype systems. In *Proceedings of the 1995 ASME International Congress and Exposition*. Citeseer, 1995.

[13] Jeff Ferrin, Robert Leishman, Randy Beard, and Tim McLain. Differential flatness based control of a rotorcraft for aggressive maneuvers. In *2011 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 2688–2693. IEEE, September 2011.

[14] Francesco Bullo and Andrew Lewis. *Geometric Control of Mechanical Systems*. Springer Science + Business Media, Inc., 2005.

[15] RL Tedrake. LQR-Trees: Feedback motion planning on sparse randomized trees. In *Robotics: Science and Systems*, Seattle, WA, 2009.

[16] SM LaValle. Rapidly-Exploring Random Trees A New Tool for Path Planning. 1998.

[17] Daniel Liberzon. *Calculus of Variations and Optimal Control Theory*. 2011.

[18] Taeyoung Lee, Melvin Leoky, and N. Harris McClamroch. Geometric tracking control of a quadrotor UAV on SE(3). In *49th IEEE Conference on Decision and Control (CDC)*, number 3, pages 5420–5425. IEEE, December 2010.

[19] Taeyoung Lee, Melvin Leok, and N.H. McClamroch. Control of Complex Maneuvers for a Quadrotor UAV using Geometric Methods on SE(3). *Asian Journal of Control*, 2011.

[20] GM Hoffmann, H Huang, and SL Waslander. Quadrotor helicopter flight dynamics and control: Theory and experiment. *American Institute of Aeronautics and Astronautics*, pages 1–20, 2007.

[21] Philippe Martin and Erwan Salaun. The true role of accelerometer feedback in quadrotor control. In *2010 IEEE International Conference on Robotics and Automation*, pages 1623–1629. IEEE, May 2010.

[22] Caitlin Powers, Daniel Mellinger, Aleksandr Kushleyev, and Bruce Kothmann. Influence of Aerodynamics and Proximity Effects in Quadrotor Flight. In *ISER 2012 (accepted)*, pages 1–14.

[23] Wayne Johnson. *Helicopter Theory*. Dover, 1994.