Path Planning and Control for a Thrust-Vectored Hopper Final Report

Rajan Aggarwal

AA203

June 6, 2021

Abstract

In this project, optimal flight controllers are designed and simulated for an acrobatic, under-actuated 'hopper' vehicle. As traditional Euler-angle representations of vehicle attitude pose singularity challenges for highly dynamic maneuvers, full quaternion-based dynamics and control were developed that better respect the group structure of SO(3). The efficacy of this framework is demonstrated on point-to-point maneuvers and a pitch flip.

1 Introduction

The authors are in the process of building a thrust-vectored 'hopper' as an experimental platform for different types of control with under-actuated dynamics (a basic schematic is shown in Figure 1). The vehicle uses an electric-ducted fan for propulsion, sitting inside a 2 degree-of-freedom servo-actuated gimbal for pitch and yaw control. Ultimately, this project seeks to develop flight controllers that: a) regulate stable hovers, b) translate the vehicle from point-to-point, and c) perform acrobatic maneuvers such as flips.

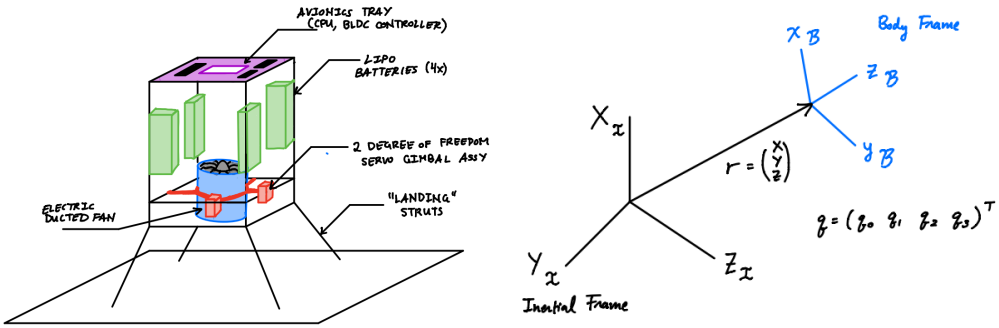

Figure 1: Left: Schematic of the hopper assembly. Total size and mass projected at ∼30x30x50 cm and 3 kg, respectively. Right: Inertial and body coordinate systems used in this report.

2 Related Work

In recent decades, there has been an outburst in development of small, agile aerial vehicles for tasks such as imaging [18], delivery [19], and entertainment [20] - [22]. This has dovetailed with a growing body of literature on flight control for these vehicles, which are especially difficult (and academically interesting) due to their nonlinear dynamics, SE(3) state space, and minimal capacity for on-board compute.

To simplify analysis, much of the literature focuses on hovering-mode control of quadcopters, which avoids attitude singularities (e.g. in [2], [23], [24], [28]). For acrobatic maneuvers, a popular framework is to leverage the differential flatness of the dynamics to plan static paths, and then use feedback control to regulate the path-tracking error dynamics [25] - [27]. These approaches can perform impressive maneuvers, but don't easily incorporate vehicle constraints and often require significant trajectory generation algorithm tuning. In this report, a simultaneous planning-and-control method is explored, which can explicitly consider constraints, incorporate SE(3) dynamics, and minimize large classes of cost functions (e.g. minimum time, control effort, jerk, etc.).

3 Dynamics

The traditional approach to describe 3D rigid body dynamics involves using Newton's 2nd law for the position states and Euler angles combined with the Newton-Euler equations for the rotational states [1] [2]. (And indeed, this is what was entertained in the midterm report.) However, Euler angles have well-known deficiencies as a representation of the group of 3D rotations SO(3): one such complication is their wrap-around near singularities, which poses difficulties for acrobatic maneuvers such as flips.

While SO(3) is a 3-parameter group, no convenient, global 3-parameter representation exists that avoids singularities [4]. A common approach is to embed the group of all rotations into a higher dimensional space (say, 4), and impose constraints that bring the group manifold dimension down to 3. The group of unit quaternions are one convenient way to do this:

$$\mathbb{S}^3 = \{q \mid \|q\|_2 = 1\},$$

where

$$q = [q_0, q_1, q_2, q_3]^T \text{ or } q_0 + q_1\hat{i} + q_2\hat{j} + q_3\hat{k}.$$

For this report, q represents the attitude of the body frame relative to the inertial frame.[1] For more information on quaternions as they pertain to rotations, the reader is referred to [5], [6] and [9]. Building off the results in [10], full quarternion-based dynamics are derived below for the vehicle.

[1] An alternate interpretation of the unit quaternion, which is far more intuitive, is its axis-angle form. Here, q is then viewed as $q = \cos\frac{\alpha}{2} + \sin(\frac{\alpha}{2})(\beta_i\hat{i} + \beta_j\hat{j} + \beta_k\hat{k})$, which represents the rotation of a coordinate system by an angle of α about the $\left(\beta_i, \beta_j, \beta_k\right)^T$ axis. In this form, α is in radians and the axis must be normalized to unit length (this maintains the unit-magnitude of q).

3.1 Translational Dynamics

The vehicle's translational dynamics are given by Newton's 2nd Law:

$$\dot{r}^{\mathcal{I}} = v^{\mathcal{I}} \tag{1}$$

$$\dot{v}^{\mathcal{I}} = \frac{1}{m} F^{\mathcal{I}}, \tag{2}$$

where r and v are the position and velocity of the center of mass in the inertial frame, and $F^{\mathcal{I}}$ is the force applied to the body in the *inertial* frame. As the applied force principally comes from thrust vectoring, which is a body-aligned wrench, the thrust vector $F^{\mathcal{B}}$ must be converted into inertial coordinates to be useful:

$$F^{\mathcal{I}} = q^* \otimes \begin{pmatrix} 0 \\ F_x^{\mathcal{B}} \\ F_y^{\mathcal{B}} \\ F_z^{\mathcal{B}} \end{pmatrix} \otimes q \tag{3}$$

The result above follows from standard equations for quaternion rotation and multiplication.[2]

3.2 Rotational Dynamics

The rotational dynamics are generally more sophisticated, but made far simpler by using quaternions:

$$\dot{q} = \frac{1}{2} q \otimes \begin{pmatrix} 0 \\ \omega^{\mathcal{B}} \end{pmatrix} \tag{4}$$

$$\dot{\omega}_{\mathcal{B}} = J^{-1}(\tau_{\mathcal{B}} - \omega_{\mathcal{B}} \times J\omega_{\mathcal{B}}), \tag{5}$$

where $\omega^{\mathcal{B}}$ is the angular velocity of the vehicle in the body frame, J is the (constant) principal inertia tensor, and $\tau^{\mathcal{B}}$ is the applied torque due to thrust-vectoring in the body frame. A derivation for Eqn 4 can be found in [6], while Eqn 5 is well-explained in [1].

3.3 Thrust Wrench Calculation

Lastly, the thrust wrench ($F^{\mathcal{B}}$, $\tau^{\mathcal{B}}$) is calculated to complete the dynamics expressions.
The force vector is obtained by first applying the rotation action of the pitch (y-axis) servo, and then the action of the yaw (z-axis) servo:

$$F^{\mathcal{B}} = \begin{pmatrix} \cos\beta & -\sin\beta & 0 \\ \sin\beta & \cos\beta & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} \cos\alpha & 0 & \sin\alpha \\ 0 & 1 & 0 \\ -\sin\alpha & 0 & \cos\alpha \end{pmatrix} \begin{pmatrix} F_t \\ 0 \\ 0 \end{pmatrix} = \begin{pmatrix} F_t \cos\alpha\cos\beta \\ F_t \cos\alpha \sin\beta \\ -F_t \sin\alpha \end{pmatrix}, \tag{6}$$

[2]Namely, if p represents the coordinates of a vector in the body frame, then $p' = q^* \otimes p \otimes q$ represents the coordinates of p in the world frame (q^* is the conjugate of q, and $\otimes$ is the quaternion product, see [5] for more details). The real components of p and p' are both zero for these calculations.

where F_t is the thrust force, α is the angle of the pitch servo, and β is the angle of the yaw servo.

The thrust torque is given by:

$$\tau^{\mathcal{B}} = r_{\text{CG to Gimbal}} \times F^{\mathcal{B}} = \begin{pmatrix} -d \\ 0 \\ 0 \end{pmatrix} \times \begin{pmatrix} F_t \cos\alpha \cos\beta \\ F_t \cos\alpha \sin\beta \\ -F_t \sin\alpha \end{pmatrix} = \begin{pmatrix} 0 \\ -F_t d \sin\alpha \\ -F_t d \cos\alpha \sin\beta \end{pmatrix} \tag{7}$$

The thrust is assumed to act through the gimbal mounts in shear at a distance of d below the center of gravity of the vehicle.

3.4 Vector Form

Grouping together the results from above, the dynamics can be expressed as:

$$\frac{d}{dt} \begin{pmatrix} r^{\mathcal{I}} \\ v^{\mathcal{I}} \\ q \\ \omega^{\mathcal{B}} \end{pmatrix} = \begin{pmatrix} v^{\mathcal{I}} \\ \frac{1}{m} q^* \otimes F^{\mathcal{B}} \otimes q \\ \frac{1}{2} q \otimes \begin{pmatrix} 0 \\ \omega^{\mathcal{B}} \end{pmatrix} \\ J^{-1}(\tau_{\mathcal{B}} - \omega_{\mathcal{B}} \times J\omega_{\mathcal{B}}) \end{pmatrix} \tag{8}$$

The state s is defined as $(r^{\mathcal{I}}\, v^{\mathcal{I}}\, q\, \omega^{\mathcal{B}})^T$, with control vector $u = (F_t, \alpha, \beta)^T$.[3]

4 Linearized Hover Regulation

Before developing acrobatic maneuvers, a hover controller is designed to robustly keep the vehicle upright during take off and landing. LQR is used for its algebraic simplicity with multi-input/multi-output systems and appealing gain and phase margins.[4]

Unfortunately, simply linearizing the quaternion dynamics about a hover equilibrium poses difficulty for pole placement, as the resulting (A, B) pair is not fully controllable (one cannot independently control all four quaternion values when rotations are inherently three dimensional) [15]. For hover regulation there is a simple solution however, which is to fall back on the Euler-angle dynamics developed in prior work, as the attitudes are not expected to be near singularities.

When linearizing those dynamics, the closed-loop hover regulator behavior is shown below (reproduced from the Midterm Report):

[3] Using s with u was, I can assure, a last resort. The standard x or z for state was taken by the position coordinates, and using q for state (as is done in Lagrangian mechanics) was taken by the quaternion vector.

[4] Specifically, an infinite-horizon policy. While slightly-suboptimal for finite horizons, the infinite-horizon policy is time invariant and thus amenable to embedded systems. The (Q, R) matrices were selected according to Bryson's Rules, which are summarized in [3].

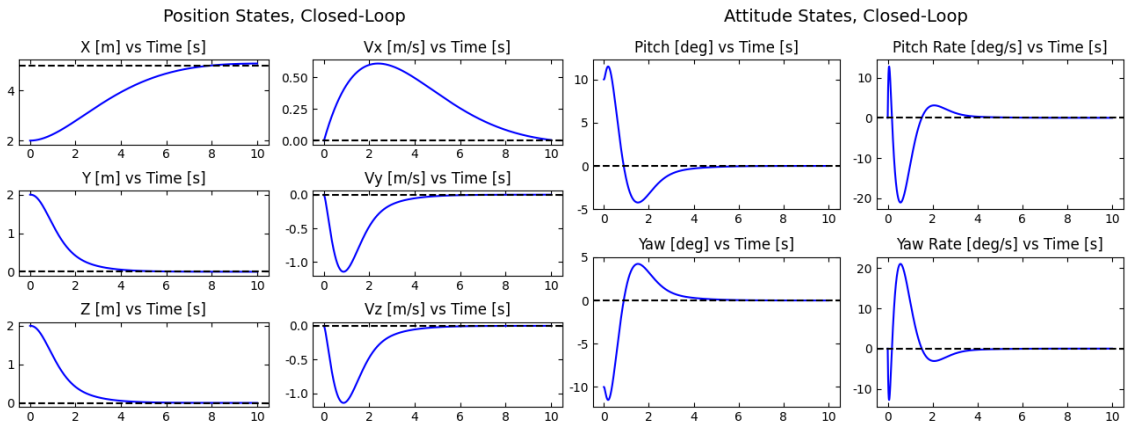

Figure 2: Position and attitude states for hover regulation from a non-equilibrium initial state.

This controller can also be used for slow translational reference signals: i.e., point-to-point movement with a joystick while maintaining hover. Ultimately, as the references become more aggressive, so do the control efforts: this may saturate the actuators leading to instability [7].

5 Point-to-Point Optimal Control

While LQR is simple to implement, it does have significant limitations for more advanced maneuvers, such as the aforementioned controller constraints. Additionally, it assumes (locally) linear dynamics, which is not always a reasonable assumption for our system as it aggressively moves through the state space.

An alternative approach is to employ direct methods for optimal control, which discretize time and solve a high-dimensional, nonlinear optimization problem whose decision variables are the control efforts, and, optionally, the state at discrete knot points. These methods are extremely powerful, and combine open-loop path planning with open-loop control at the expense of computational burden and returning a static policy. There are many variations on this theme[5], but for the purposes of this project a simple direct method is used, that discretizes both the states and controls and solves the resulting optimization problem. For point-to-point movement, this results in the following:

[5]For instance: iterative LQR (iLQR) methods iteratively 'shape' the trajectory through the state space and re-linearize/re-quadratize the dynamics and cost to enable the use of LQR-like dynamic programming for non-linear model optimizations; sequential convex programming (SCP) does similar sequential convexification of the optimal control problem but critically does not need a feasible initial trajectory; and model-predictive control iteratively applies open-loop actions, however they are found, in a receding-horizon fashion to imbue some closed-loop robustness.

$$
\begin{aligned}
\underset{s_0,\ldots,s_N,u_0,\ldots,u_{N-1}}{\text{minimize}} \quad & (s_N - s_{\text{final}})^T Q (s_N - s_{\text{final}}) \\
& + \sum_{k=0}^{N-1} (s_k - s_{\text{final}})^T Q (s_k - s_{\text{final}}) + (u_k - u_{\text{eq}})^T R (u_k - u_{\text{eq}}) \\
\text{subject to} \quad & s_{k+1} = f_d(s_k, u_k) \\
& -\alpha_{\max} \leq \alpha_k \leq \alpha_{\max} \\
& -\beta_{\max} \leq \beta_k \leq \beta_{\max} \\
& F_{t,\min} \leq F_{t,k} \leq F_{t,\max} \\
& x_k \geq 0 \\
& s_0 = s_{\text{init}}
\end{aligned}
$$

The cost function is a standard LQR-like quadratic cost on the state and control effort. The constraints are, respectively: Euler-discretized nonlinear dynamics evolution[6], thrust vector servo angle limits, thrust magnitude limits, vehicle altitude above the ground ($x \geq 0$m), and state initialization.

As the dynamics are nonlinear, the overall problem is nonconvex. Sequential convex programming is one approach to handle problems like these, but for the purposes of this report, an interior-point nonconvex optimization software was used (IPOPT supported by CasADi auto-differentiation) [11] [12] [13].

The horizon for movement from $(1, 0, 0)$ m to $(5, 2, -2)$ m is shown below:

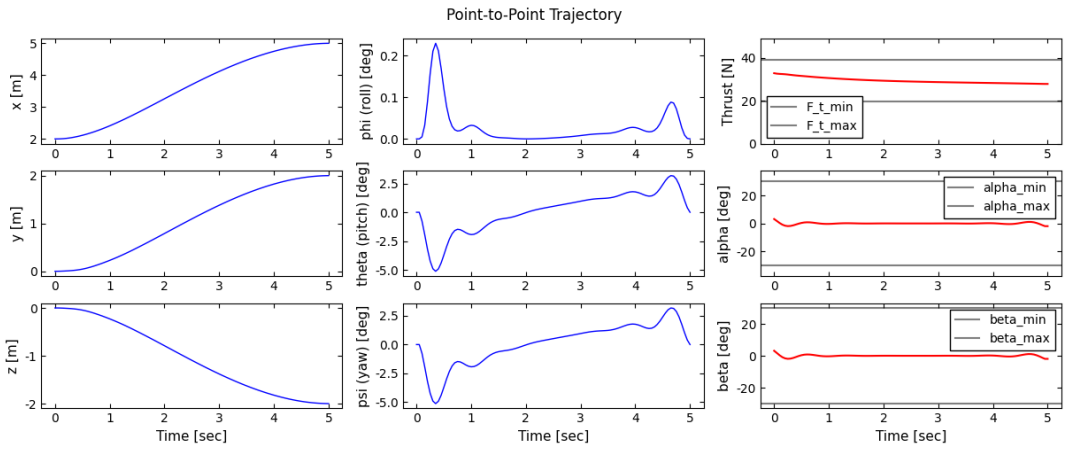

Figure 3: Point-to-point position and attitude states over a 5 sec horizon (N = 100, dt = .05 sec). Boundary condition states are hover equilibria. Video link on Google Drive (click).

[6] After an Euler-step, the quaternion is re-normalized to unit length, to maintain its representation of a rotation. This is an approach taken with success in [16], but next steps would include the use of variational integrators which explicitly respect the group structure of quaternions and preserve the momentum- and energy- quantities of the dynamics [17].

In reality, executing the open-loop controls alone will not work due to model mismatches and disturbances – at the very least, the open loop actions should be cascaded as is done in an MPC. For enhanced robustness or reduced computational load, a locally-linear LQR tracking controller can be used to improve the closed-loop performance.

6 Pitch Flip

To better demonstrate the versatility of quaternion-dynamics and direct methods for optimal control, a pitch flip is solved. Simply constraining the start and end states of the optimization to be upright hover equilibria is not enough to 'nudge' the optimization into doing a flip: for this, a waypoint is added for an upside-down orientation and is costed in the objective function.

$$
\begin{aligned}
\underset{s_0,\ldots,s_N,u_0,\ldots,u_{N-1}}{\text{minimize}} \quad & \sum_{(w,k)\in\mathcal{W}} 1 - |q_k^T w| \\
& + \sum_{k=0}^{N-1} (u_k - u_{\text{eq}})^T R (u_k - u_{\text{eq}}) \\
\text{subject to} \quad & s_{k+1} = f_d(s_k, u_k) \\
& -\alpha_{\max} \le \alpha_k \le \alpha_{\max} \\
& -\beta_{\max} \le \beta_k \le \beta_{\max} \\
& F_{t,\min} \le F_{t,k} \le F_{t,\max} \\
& x_k \ge 0 \\
& s_0 = s_{\text{init}}
\end{aligned}
$$

The set $\mathcal{W}$ contains (w, k) tuples that describe the quaternion attitude w of a waypoint at an index k down the horizon, and was inspired by [14]. As the group of unit quaternions is not closed under subtraction, the dot product is used to assess how similar one rotation is to another [8].

The pitch flip state and control tapes are shown below:

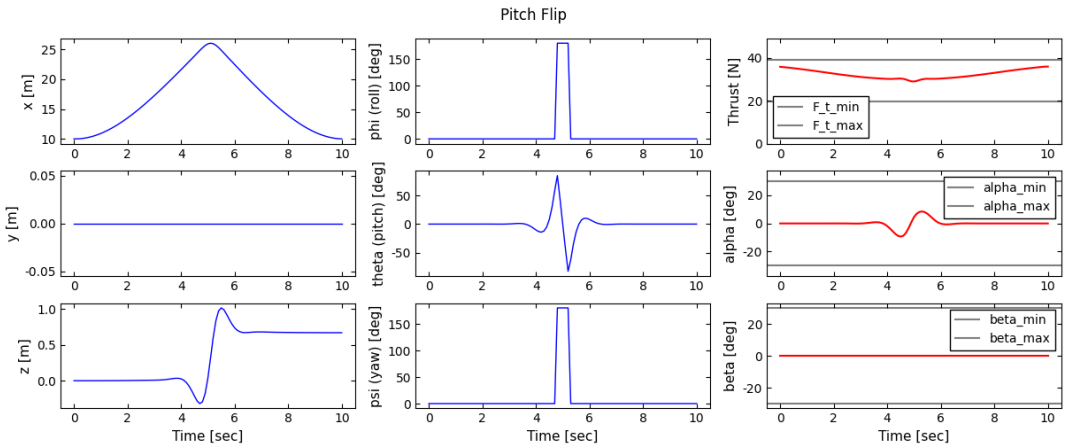

Figure 4: Position and attitude states for a pitch flip. Video link on Google Drive (click).

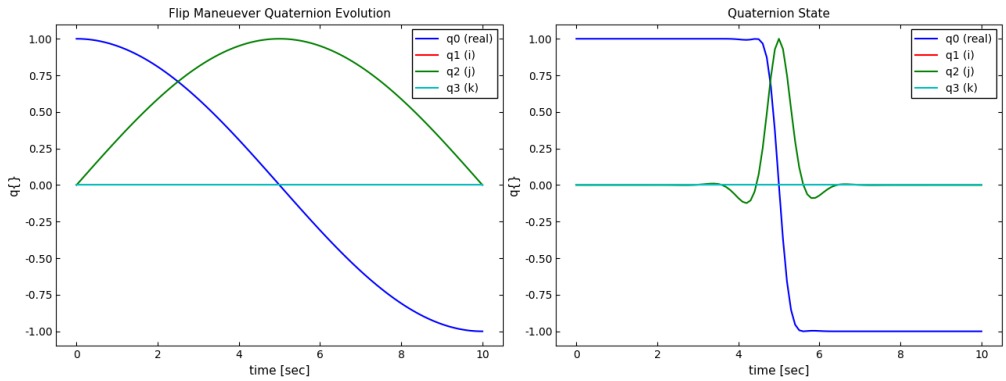

Figure 5: Left: Reference quaternion trajectory for the flip, and Right: Actual quaternion trajectory. Only 3 waypoints were used in this optimization, at the beginning, middle and end of the horizon.

7 Discussion and Next Steps

In the above sections, quaternion-dynamics and control were developed for hover regulation, point-to-point movement, and a pitch flip. There are many avenues for future work, but a few salient ones are listed below:

- Develop tracking controllers for offline-designed trajectories. This would require a state estimator but would result in robust closed-loop tracking performance, without the need for re-solving new trajectories online as in MPC.
- Upgrade the integration scheme in trajectory generation to be geometric à la [17]. This would more faithfully preserve the group structure than the current augmented Euler steps.
- Explore ways to convexify the trajectory optimization, to improve solve times and prepare the optimization for embedded integration. Look into 'lossless' convexification techniques (e.g. in [29]): one approach could be to plan in the body-wrench space instead of the low-level control space.
- After making the preceding improvements to the trajectory optimization, solve for simultaneous pitch and yaw flips.
- Generally, begin translating some of these controllers into embedded code for testing (start off with LQR hover and an LQR point-to-point, which are static, linear policies).
- Investigate the effect of gyro dynamics with the electric-ducted fan, and whether or not this effect can be used to regulate vehicle roll (there is no direct roll control actuator).

The accompanying YouTube video for this report is provided here: link.

References

[1] Donald Greenwood, "Principles of Dynamics," Prentice Hall, 2nd ed. (1988).

[2] T. Luukkonen, "Modelling and control of a quadcopter," Independent research project in applied mathematics, Espoo (2011).

[3] G. Franklin, J. D. Powell and M. Workman, "Digital Control of Dynamic Systems," Addison-Wesley Publishing Co., 2nd ed. (1990).

[4] J. Stuelpnagel, "On the parameterization of the three-dimensional rotation group" 4 *SIAM Review* 6 (1964).

[5] J. Solà, "Quaternion kinematics for the error-state Kalman filter," Course Notes, Institut de Robòtica i Informàtica Industrial (2017).

[6] B. Graf, "Quaternions and Dynamics," Course Notes, École polytechnique fédérale de Lausanne (2007).

[7] C. Jones, F. Borrelli, M. Morari, "Model Predictive Control," Course Notes, ETH Zürich and UC Berkeley (2015).

[8] J. J. Kuffner, "Effective Sampling and Distance Metrics for 3D Rigid Body Path Planning," *IEEE International Conference on Robotics and Automation* (2004).

[9] Y.-B. Jia, "Quaternions and Rotations," Course Notes, Iowa State University (2013).

[10] E. Fresk and G. Nikolakopoulos, "Full Quaternion Based Attitude Control for a Quadrotor," *European Control Conference* (2013).

[11] A. Wächter and L. T. Biegler, "On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming" 1 *Mathematical Programming* 106 (2006).

[12] A. Wächter, "Short tutorial: Getting started with Ipopt in 90 minutes," *Combinatorial Scientific Computing* (2009).

[13] J. Andersson et al., "CasADi – A software framework for nonlinear optimization and optimal control," 1 *Mathematical Programming Computation* 11 (2019).

[14] B. Jackson, K. Tracy and Z. Manchester, "Planning with Attitude," *IEEE Robotics and Automation Letters, Manuscript* (2020).

[15] L. Lustosa et al., "A new look at the uncontrollable linearized quaternion dynamics with implications to LQR design in underactuated systems," *European Control Conference* (2018).

[16] M. Andrle and J. Crassidis, "Geometric Integration of Quaternions," 6 *Journal of Guidance, Control and Dynamics* 36 (2013).

[17] T. Lee, N. H. McClamroch and M. Leok, "Optimal Control of a Rigid Body using Geometrically Exact Computations on SE(3)," *IEEE Conference on Decision & Control* (2006).

[18] Skydio Autonomous Drones, URL https://www.skydio.com/ (last accessed 16 April 2021)

[19] Amazon Prime Air, URL: https://www.amazon.com/Amazon-Prime-Air/b?ie=UTF8node=8037720011 (last accessed 16 April 2021)

[20] Intel Light Show, URL: https://www.intel.com/content/www/us/en/technology-innovation/aerial-technology-light-show.html (last accessed 16 April 2021)

[21] Drone Racing League, URL: https://thedroneracingleague.com/ (last accessed 16 April 2021)

[22] J. Delmerico et al., "Are We Ready for Autonomous Drone Racing? The UZH-FPV Drone Racing Dataset," International Conference on Robotics and Automation (ICRA), Montreal, Canada (2019).

[23] M. Hehn and R. D'Andrea, "Quadcopter Trajectory Generation and Control," 44 IFAC Proceedings Volumes 1 (2011).

[24] A. Richards et al., "Coordination and Control of Multiple UAVs," AIAA Guidance, Navigation and Control Conference and Exhibit, Monterey, CA, USA (2002).

[25] D. Mellinger and V. Kumar, "Minimum Snap Trajectory Generation and Control for Quadcopters," *IEEE International Conference on Robotics and Automation* (2011).

[26] D. Mellinger, N. Michael and V. Kumar, "Trajectory generation and control for precise aggressive maneuvers with quadrotors" 5 *The International Journal of Robotics Research* 31 (2012).

[27] T. Lee, M. Leok and N. H. McClamroch, "Geometric Tracking Control of a Quadrotor UAV on SE(3)," 49th IEEE Conference on Decision and Control, Atlanta, GA, USA (2010).

[28] P. Foehn and D. Scaramuzza, "Onboard State Dependent LQR for Agile Quadrotors," *IEEE International Conference on Robotics and Automation* (2018).

[29] B. Açikmeşe, J. Carson III and L. Blackmore, "Lossless Convexification of Nonconvex Control Bound and Pointing Constraints of the Soft Landing Optimal Control Problem," 6 *IEEE Transactions on Control Systems Technology* 21 (2013).