Planning and Control with Attitude

Brian Jackson and Zachary Manchester

Stanford University, 496 Lomita Mall, Stanford, CA, USA,
{bjack205,zacm}@stanford.edu

Abstract. Planning and controlling trajectories for floating-base robotic systems that experience large attitude changes is challenging due to the nontrivial group structure of 3D rotations. This paper introduces an accessible and unified approach for tracking control and optimization-based planning on the space of rotations based on vector calculus and linear algebra. The methodology is used to derive an extension of the Linear-Quadratic Regulator (LQR) to systems with arbitrary attitudes, which we call Multiplicative LQR (MLQR). We compare MLQR to a state-of-the-art geometric tracking controller designed specifically for quadrotors and demonstrate superior tracking and robustness. An iterative variant of MLQR is compared to existing methods on useveral constrained trajectory optimization problems, again demonstrating improved runtime performance.

Keywords: motion planning and control, quaternions, optimal control, linear quadratic regulator

1 Introduction

Many useful robotic systems—including quadrotors, airplanes, satellites, autonomous underwater vehicles, and quadrupeds—can perform arbitrarily large three-dimensional translations and rotations as part of their normal operation. While representing translations is straightforward and intuitive, effectively representing the nontrivial group structure of 3D rotations has been a topic of study for many decades. Although we can intuitively deduce that rotations are three-dimensional, a globally non-singular three-parameter representation of the space of rotations does not exist [26]. As a result, when parameterizing rotations, we must either a) pick a three-parameter representation and deal with discontinuities, or b) pick a higher-dimensional representation and deal with constraints between the parameters. While simply representing attitude is nontrivial, generating and tracking motion plans for floating-base systems is an even more challenging problem.

Early work on control problems involving the rotation group dates back to the 1970s, with extensions of linear control theory to spheres [4] and $SO(3)$ [3]. Effective attitude tracking controllers have been developed for satellites [30], quadrotors [8, 18, 17, 12, 28, 21], and a 3D inverted pendulum [5] using various methods for calculating three-parameter attitude errors.

More recently, these ideas have been extended to trajectory generation [33], sample-based motion planning [34, 15], and optimal control. Approaches to optimal control on attitude problems include analytical methods applied to satellites [25], discrete mechanics [14, 13, 16], a combination of sampling-based planning and constrained trajectory optimization for satellite formations [9, 2], projection operators [23], or more general theory for optimization on manifolds [29]. Nearly all of these methods rely heavily on principles from differential geometry and Lie group theory; however, despite these works, many recent papers in the robotics community continue to apply traditional methods for motion planning and control with no regard for the group structure of rigid body motion [1, 6, 31, 10].

In this paper, we make a departure from previous approaches to geometric planning and control that rely heavily on ideas and notation from differential geometry, and instead use only basic mathematical tools from linear algebra and calculus that should be familiar to most roboticists. Similar to [19, 32], in Sec. 3 we introduce a quaternion differential calculus, but take a significantly simpler and more general approach that unifies both planning and control, enabling straight-forward adaptation of existing algorithms to systems with quaternion states. To make this concrete, in Sec. 4 we apply our approach to derive an extension to the linear-quadratic regulator (LQR) that we call multiplicative LQR (MLQR), which is the control dual to the multiplicative extended Kalman filter (MEKF) from the state estimation literature [20]. In Sec. 5 we provide several simulation results demonstrating the application of our method to tracking control and constrained trajectory optimization. In summary, our contributions include:

- A unified approach to quaternion differential calculus entirely based on standard vector calculus and linear algebraic operations
- Derivation of multiplicative LQR, an adaptation of LQR to the control of systems with quaternion states
- A set of benchmark motion-planning and trajectory-tracking problems in which we compare our approach to existing methods commonly used in the robotics community to account for 3D rotations

2 Background

We begin by defining some useful conventions and notation. Attitude is defined as the rotation from the robot's body frame to a global inertial frame. We also define gradients to be row vectors, that is, for $f(x) : \mathbb{R}^n \to \mathbb{R}$, $\frac{\partial f}{\partial x} \in \mathbb{R}^{1\times n}$.

2.1 Unit Quaternions

We leverage the fact that quaternions are linear operators and that the space of quaternions $\mathbb{H}$ is isomorphic to $\mathbb{R}^4$ to explicitly represent—following the Hamilton convention—a quaternion $\mathbf{q} \in \mathbb{H}$ as a standard vector $q \in \mathbb{R}^4 := [q_s \;\; q_v^T]^T$

where $q_s \in \mathbb{R}$ and $q_v \in \mathbb{R}^3$ are the scalar and vector part of the quaternion, respectively.

Quaternion multiplication is defined as

$$\mathbf{q}_2 \otimes \mathbf{q}_1 = L(q_2)q_1 = R(q_1)q_2 \tag{1}$$

where $L(q)$ and $R(q)$ are orthonormal matrices defined as

$$L(q) := \begin{bmatrix} q_s & -q_v^T \\ q_v & q_s I + [q_v]^\times \end{bmatrix} \tag{2}$$

$$R(q) := \begin{bmatrix} q_s & -q_v^T \\ q_v & q_s I - [q_v]^\times \end{bmatrix}, \tag{3}$$

and $[x]^\times$ is the skew-symmetric matrix operator

$$[x]^\times := \begin{bmatrix} 0 & -x_3 & x_2 \\ x_3 & 0 & -x_1 \\ -x_2 & x_1 & 0 \end{bmatrix}. \tag{4}$$

The inverse of a unit quaternion q^{-1}, giving the opposite rotation, is equal to its conjugate q^*, which is simply the same quaternion with a negated vector part:

$$\mathbf{q}^* = Tq := \begin{bmatrix} 1 & \\ & -I_3 \end{bmatrix} q \tag{5}$$

The following identities, which are easily derived from (2)–(5), are extremely useful:

$$L(Tq) = L(q)^T = L(q)^{-1} \tag{6}$$

$$R(Tq) = R(q)^T = R(q)^{-1}. \tag{7}$$

We will sometimes find it helpful to create a quaternion with zero scalar part from a vector $r \in \mathbb{R}^3$. We denote this operation as,

$$\hat{r} = Hr \equiv \begin{bmatrix} 0 \\ I_3 \end{bmatrix} r. \tag{8}$$

Unit quaternions rotate a vector through the operation $\hat{r}' = \mathbf{q} \otimes \hat{r} \otimes \mathbf{q}^*$. This can be equivalently expressed using matrix multiplication as

$$r' = H^T L(q) R(q)^T H r = A(q) r, \tag{9}$$

where $A(q)$ is the rotation matrix in terms of the elements of the quaternion [20].

2.2 Rigid Body Dynamics

In the current work we will restrict our focus to rigid bodies moving freely in 3D space. That is, we consider systems with dynamics of the following form:

$$x = \begin{bmatrix} r \\ R \\ v \\ \omega \end{bmatrix}, \quad \dot{x} = \begin{bmatrix} v \\ \texttt{kinematics}(R, \omega) \\ \frac{1}{m} F_G(x, u) \\ J^{-1}(\tau_L(x, u) - \omega \times J\omega) \end{bmatrix} \tag{10}$$

where x and u are the state and control vectors, $r \in \mathbb{R}^3$ is the position, $R \in SO(3)$ is the attitude, $v \in \mathbb{R}^3$ is the linear velocity, and $\omega \in \mathbb{R}^3$ is the angular velocity. $m \in \mathbb{R}$ is the mass, $J \in \mathbb{R}^{3\times3}$ is the inertia matrix, $F_G(x, u) \in \mathbb{R}^3$ are the forces in the global frame, $\tau_L(x, u)$ are the moments in the local (body) frame, and `kinematics` are the kinematics for the chosen attitude representation. The kinematics for unit quaternions are

$$\dot{q} = \frac{1}{2}\mathbf{q} \otimes \hat{\omega} = \frac{1}{2} L(q) H \omega. \tag{11}$$

3 Quaternion Differential Calculus

We now present a simple but powerful method for taking derivatives of functions involving quaternions based on the notation and linear algebraic operations outlined in Sec. 2.1.

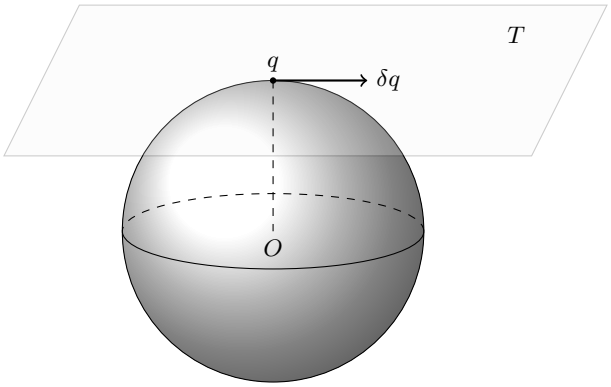


Fig. 1: When linearizing about a point q on an sphere $\mathbb{S}^{n-1}$ in n-dimensional space, the tangent space T is a plane living in $\mathbb{R}^{n-1}$, illustrated here with $n = 3$. Therefore, when linearizing about a unit quaternion $q \in \mathbb{S}^3$, the space of differential rotations lives in $\mathbb{R}^3$.

Derivatives consider the effect an infinitesimal perturbation to the input has on an infinitesimal perturbation to the output. For vector spaces, the composition of the perturbation with the nominal value is simple addition and the

infinitesimal perturbation lives in the same space as the original vector. For unit quaternions, however, neither of these are true; instead, they compose according to (1), and infinitesimal unit quaternions are (to first order) confined to a 3-dimensional plane tangent to $\mathbb{S}^3$ (see Fig. 1). The fact that differential unit quaternions are three-dimensional should make intuitive sense: Rotations are inherently three-dimensional and differential rotations should live in the same space as angular velocity, i.e. $\mathbb{R}^3$.

There are many possible three-parameter representations for small rotations in the literature. Many authors use the exponential map [3, 33, 16, 23, 24, 7, 29], while others have used the Cayley map (also known as Rodrigues parameters) [14, 13], Modified Rodrigues Parameters (MRPs) [27], or the vector part of the quaternion [8]. We choose Rodrigues parameters [20] because they are computationally efficient and do not inherit the sign ambiguity associated with unit quaternions. The mapping between a vector of Rodrigues parameters $\phi \in \mathbb{R}^3$ and a unit quaternion q is known as the Cayley map:

$$q = \varphi(\phi) = \frac{1}{\sqrt{1 + \|\phi\|^2}} \begin{bmatrix} 1 \\ \phi \end{bmatrix}. \tag{12}$$

We will also make use of the inverse Cayley map:

$$\phi = \varphi^{-1}(q) = \frac{q_v}{q_s}. \tag{13}$$

3.1 Jacobian of Vector-Valued Functions

When taking derivatives with respect to quaternions, we must take into account both the composition rule and the nonlinear mapping between the space of unit quaternions and our chosen three-parameter error representation. Let $\phi \in \mathbb{R}^3$ be a differential rotation applied to a function with quaternion inputs $y = h(q) : \mathbb{S}^3 \to \mathbb{R}^p$, such that

$$y + \delta y = h(L(q)\varphi(\phi)) \approx h(q) + \nabla h(q)\phi. \tag{14}$$

We can calculate the Jacobian $\nabla h(q) \in \mathbb{R}^{p\times 3}$ by differentiating (14) with respect to ϕ, evaluated at $\phi = 0$:

$$\nabla h(q) = \frac{\partial h}{\partial q} L(q) H := \frac{\partial h}{\partial q} G(q) = \frac{\partial h}{\partial q} \begin{bmatrix} -q_v^T \\ sI_3 + [q_v]^\times \end{bmatrix} \tag{15}$$

where $G(q) \in \mathbb{R}^{4\times 3}$ is the *attitude Jacobian*, which essentially becomes a "conversion factor" allowing us to apply results from standard vector calculus to the space of unit quaternions. This form is particularly useful in practice since $\partial h/\partial q \in \mathbb{R}^{p\times 4}$ can be obtained using finite difference or automatic differentiation. As an aside, although we have used Rodrigues parameters, $G(q)$ is actually the same (up to a constant scaling factor) for any choice of three-parameter attitude representation.

3.2 Hessian of Scalar-Valued Functions

If the output of h is a scalar ($p = 1$), then we can find its Hessian by taking the Jacobian of (15) with respect to ϕ using the product rule, again evaluated at $\phi = 0$:

$$\nabla^2 h(q) = G(q)^T \frac{\partial^2 h}{\partial q^2} G(q) + I_3 \frac{\partial h}{\partial q} q, \tag{16}$$

where the second term comes from the second derivative of $\varphi(\phi)$. Similar to $G(q)$, this ends up being the same (up to a scaling factor) for any choice of three-parameter attitude representation.

3.3 Jacobian of Quaternion-Valued Functions

We now consider the case of a function that maps unit quaternions to unit quaternions, $q' = f(q) : \mathbb{S}^3 \to \mathbb{S}^3$. Here we must also consider the non-trivial effect of a differential value applied to the output, i.e.:

$$L(q')\varphi(\phi') = f(L(q)\varphi(\phi)). \tag{17}$$

Solving (17) for ϕ' we find,

$$\phi' = \varphi^{-1}\left(L(q')^T f(L(q)\varphi(\phi))\right) \approx \nabla f(q)\, \phi. \tag{18}$$

Finally, the desired Jacobian is obtained by taking the derivative of (18) with respect to ϕ:

$$\nabla f(q) = H^T L(q')^T \frac{\partial f}{\partial q} L(q) H = G(q')^T \frac{\partial f}{\partial q} G(q). \tag{19}$$

The leading $G(q')^T$ comes from the fact that as $\phi' \to 0$, $L(q')f(q) \to I_q$, where I_q is the quaternion identity. Differentiating through the inverse map, evaluated at the quaternion identity, we find that $\partial\varphi^{-1}/\partial q \to H^T$ for any three-parameter attitude representation.

4 Multiplicative LQR

Leveraging the methods from the previous section, we derive multiplicative LQR (MLQR), a variant of LQR that correctly accounts for the group structure of rotations. For concreteness, we consider a system with rigid body dynamics, as presented in Sec. 2.2, and design a controller to stabilize the system about a dynamically feasible discretized reference trajectory $\bar{x}_{0:N}, \bar{u}_{0:N}$ with N time steps. We begin by linearizing the dynamics about the reference trajectory using (19). Our linearized error dynamics become

$$\delta x_{k+1} = A_k \delta x_k + B_k \delta u_k \tag{20}$$

where

$$A_k = E(\bar{x}_{k+1})^T \frac{\partial f}{\partial x}|_{\bar{x}_k,\bar{u}_k} E(\bar{x}_k), \quad B_k = E(\bar{x}_{k+1})^T \frac{\partial f}{\partial u}|_{\bar{x}_k,\bar{u}_k}, \tag{21}$$

and $\delta x_k \in \mathbb{R}^{12}$ and $E(x_k) \in \mathbb{R}^{12\times 13}$ are the state error and state error Jacobian:

$$\delta x_k = \begin{bmatrix} r_k - \bar{r}_k \\ \varphi^{-1}(\bar{\mathbf{q}}_k^{-1} \otimes \mathbf{q}_k) \\ v_k - \bar{v}_k \\ \omega_k - \bar{\omega}_k \end{bmatrix}, \; E(x) = \begin{bmatrix} I_3 & & & \\ & G(q) & & \\ & & I_3 & \\ & & & I_3 \end{bmatrix}. \tag{22}$$

With our linearized system, we can apply the traditional LQR Riccati recursion,

$$P_k = W_k + A_k^T P_{k+1} A_k - A_k^T P_{k+1}^T B_k (R_k + B_k^T P_{k-1} B_k)^{-1} B_k^T P_{k+1} A_k, \tag{23}$$

which minimizes the quadratic cost function

$$\frac{1}{2} \sum_{k=0}^{N} \delta x_k^T Q_k \delta x_k + \delta u_k^T R_k \delta u_k \tag{24}$$

of the state *error*, where $Q_k \in \mathbb{R}^{12\times 12}$ is the weight matrix on the state *error*, and $R_k \in \mathbb{R}^m$ is the weight matrix on the controls. From this, we get our non-linear feedback policy

$$u = K_k \delta x_k + \bar{u}_k. \tag{25}$$

where δx_k is computed online using the non-linear error function (22) and the linear feedback gain K_k is calculated using the standard LQR formula:

$$K_k := -(R_k + B_k^T P_{k+1} B_k)^{-1} B_k^T P_{k+1} A_k. \tag{26}$$

The multiplicative Linear Quadratic Regulator is summarized as follows:

1. Pick $Q_k \in \mathbb{R}^{12\times 12}$ to weight the state *error*, and $R_k \in \mathbb{R}^{m\times m}$
2. Linearize the dynamics using the state error Jacobian (22)
3. Compute K_k using the standard LQR Riccati recursion
4. Online, compute the control using (22) and (25).

4.1 Constrained Iterative MLQR

As demonstrated above, it is very straight-forward to adapt LQR to use quaternions. We similarly use this technique to adapt nonlinear trajectory optimization algorithms. Here we present the modifications to the ALTRO solver [11], which uses iterative LQR (iLQR) combined with an augmented Lagrangian approach to handle constraints.

For each iteration of iLQR within ALTRO, we can calculate a quadratic approximation of the cost function using (15) and (16). Identical to (20), we

use (19) to linearize the dynamics and use MLQR to solve for the feedback and feedforward gains.

The only other modification to the algorithm is during the iLQR "forward pass" that simulates the system with the closed-loop MLQR controller, where we use (22) to calculate the nonlinear feedback policy during the forward roll-outs. For robustness, we may also add an additional criterion to the line search that checks for singularities in the three-parameter error state which, for the Cayley map, occur at 180°, at which point the step size should be reduced.

4.2 Quaternion Cost Functions

In addition to the straight-forward modifications to the LQR algorithm itself, we need to carefully consider the types of cost functions we minimize. We frequently minimize costs that penalize distance from a goal state, e.g. $\frac{1}{2}(x-x_g)^T Q(x-x_g)$; however, naïve substraction of unit quaternions is ill-defined. We propose two different cost functions that accomplish similar behavior. For sake of clarity and space, we only consider the costs on the quaternion variables: costs on the other states and the control variables remain unaffected.

Error Quadratic Rather than simple subtraction, we can use a quadratic function on the three-parameter error state (22):

$$J_{\text{err}} = \frac{1}{2}\phi^T Q \phi = \frac{1}{2}\left(\varphi^{-1}(\delta q)\right)^T Q \left(\varphi^{-1}(\delta q)\right). \tag{27}$$

where $\delta q = L(q_g)^T q$, and $\phi = \varphi \delta q$. The gradient and Hessian of (27) are

$$\nabla J_{\text{err}} = \phi^T Q D(\delta q) G(\delta q) \tag{28}$$

$$\nabla^2 J_{\text{err}} = G(\delta q)^T \left(D(\delta q)^T Q D(\delta q) + \nabla D\right) G(\delta q) + I_3(\phi^T Q D(\delta q))\delta q \tag{29}$$

where, for the Cayley map,

$$D(q) = \frac{\partial \varphi^{-1}}{\partial q} = -\frac{1}{q_s^2}\begin{bmatrix} q_v & -\frac{1}{q_s} I_3 \end{bmatrix} \tag{30}$$

$$\nabla D = \frac{\partial}{\partial q}\left(D(q)^T Q \phi\right) = -\frac{1}{q_s^2}\begin{bmatrix} -2\frac{q_v}{q_s}^T Q\phi & \phi^T Q \\ Q\phi & 0 \end{bmatrix}. \tag{31}$$

Geodesic Distance Alternatively, we can use the geodesic distance between two quaternions [15],

$$J_{\text{geo}} = (1 - |q_g^T q|), \tag{32}$$

whose gradient and Hessian are,

$$\nabla J_{\text{geo}} = \pm q_g^T G(q) \tag{33}$$

$$\nabla^2 J_{\text{geo}} = \pm I_3 q_g^T q, \tag{34}$$

where the sign of the Hessian corresponds to the sign of $q_d^T q$.

5 Experiments

Code code for all experiments is available on GitHub at \texttt{https://github.com/RoboticExplorationLab/PlanningWithAttitude}.

5.1 Trajectory Optimization

In this section we present several trajectory optimization problems solved using our modified version of ALTRO. The two quadrotor examples use 101 knot points for 5 second trajectories, and the airplane example uses 51 knot points for a 1.25 second trajectory. All results were run on a desktop computer with an Intel i9-9900 3.10 GHz processor with 32GB of memory. Timing results for all three experiments are given in Fig. 2.

In all examples, MLQR is implemented with a unit quaternion attitude state and Rodrigues parameters as the error state. The first example compares the two cost functions given in Sec. 4.2, while the last two use the geodesic quaternion cost and compare iterative MLQR (iMLQR) from Sec. 4 to iLQR using roll-pitch-yaw Euler angles following the convention from [22], and quaternions (treating them naïvely as vectors). All examples re-normalize the quaternion in the dynamics function and use a third-order explicit Runge-Kutta integrator.

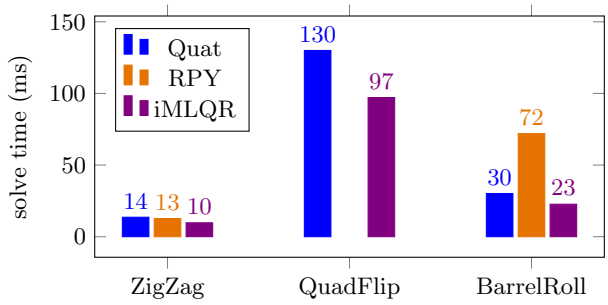
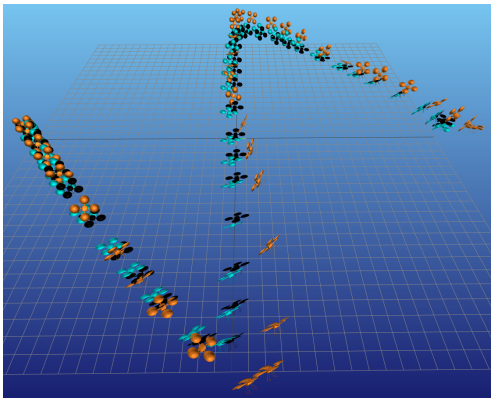


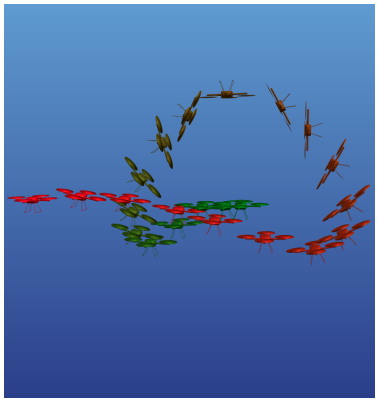
Fig. 2: Timing results for the airplane barrell roll, quadrotor flip, and quadrotor zig-zag examples using iLQR with roll-pitch-yaw Euler angles (RPY), quaternions, and iMLQR. The timing result for the Quadrotor flip with Euler angles is omitted since it failed to converge.

Quadrotor Zig-Zag We set up a fairly simple trajectory optimization problem to achieve a "zig-zag" pattern, as shown in Fig. 3a using cost funtions of the form

$$\frac{1}{2}\sum_{k\in\mathcal{N}} \ell(x_k, x_{\text{nom}}, Q_{\text{nom}}) + \frac{1}{2}\sum_{k\in\mathcal{W}} \ell(x_k, \bar{x}_k, Q_w) + \frac{1}{2}\sum_{k=0} u_k^T R u_k \tag{35}$$



(a) Zig-zag trajectory. The black quadrotors indicate the reference trajectory generated by iMLQR. The yellow quadrotors track the reference trajectory using time-varying MLQR, while the red uses the HFCA controller from [28].



(b) Snapshots of the quadrotor flip trajectory. The red-colored quadrotors represent the state near t=0 s and the green-colored quadrotors represent the state near t=5.0 s

Fig. 3: Examples applying MLQR to trajectory optimization on quadrotors

with $\mathcal{W} = \{33, 66, 101\}$, $\mathcal{N} = \{1 : 101\} \setminus \mathcal{W}$, and the waypoint positions were $(10, 0, 1)$,$(-10, 0, 1)$ and $(0, 10, 0)$. We compared the two cost functions from Sec. 4.2, using a standard quadratic penalty on the non-quaternion states.

$$\ell_{\text{err}}(x, \bar{x}, Q) = \delta x^T Q \delta x \tag{36}$$

$$\ell_{\text{geo}}(x, \bar{x}, Q) = (x - \bar{x})^T \bar{Q} (x - \bar{x}) + w(1 \pm \bar{q}^T q) \tag{37}$$

where $\bar{Q} = \text{diag}(Q_r, \mathbf{0}_4, Q_v, Q_\omega)$, where Q_r, Q_v, Q_ω are the weights of Q for position, and linear and angular velocity. We used the following weighting matrices: $Q_{\text{nom}} = \text{diag}(10^{-5} I_6, 10^{-3} I_6)$, $Q_w = \text{diag}(10^3 I_3, I_9)$, $Q_N = \text{diag}(10 I_3, 100 I_3, 10 I_6)$. For the geodesic weights w, we used 0, 1, and 10 for the nominal, waypoint, and terminal cost functions, respectively.

The geodesic cost function performed significantly better than the error quadratic, converging in 10 ms and 13 iterations, compared to 53 ms in 29 iterations. In addition to better convergence behavior, the geodesic cost is about half as expensive to compute, which is unsurprising when comparing the gradients and Hessians in Sec. 4.2.

Quadrotor Flip Similar to the previous example, we use a set of waypoints to encourage the quadrotor to do a flip. The cost weights for this example were

$$\begin{aligned} Q_{\text{nom}} &= \text{diag}(10^{-2}, 10^{-2}, 5 \times 10^{-2}, 10^{-3} I_3, 10^{-3} I_3, 10^{-2} I_3) \\ Q_w &= \text{diag}(10^3, 10, 10^3, 5000 I_3, I_3, 10 I_3) \\ Q_N &= \text{diag}(10 I_3, 100 I_3, 10 I_6) \end{aligned}$$

Four intermediary knotpoints were used to encourage the quadrotor to be at angles of 90°, 180°, 270°, and 360°. The quadrotor was constrained to stay above the floor and move to a goal state 2 meters away in the $+y$ direction. The solver is initialized with a dynamically infeasible trajectory that linearly interpolates between the initial and final states, rotating the quad around the x-axis a full 360°.

Figure 3b shows snapshots of the trajectory as generated using iMLQR within the ALTRO solver to handle constraints. The trajectory generated by iLQR with quaternions and quadratic costs was successful but had a slightly different trajectory. iLQR with Euler angles failed to solve the problem.

5.2 Airplane Barrel Roll

An airplane model with aerodynamic coefficients fit from real wind-tunnel data is tasked to do a barrel roll by setting a high terminal cost for being upside-down, see Fig. 4. The solver is initialized with level flight trim conditions. For both the quadrotor flip and the airplane barrel roll, iMLQR converged faster than the pure quaternion version. Despite the extra matrix multiplications and extra Hessian term, it also was faster per iteration than its iLQR counterpart. For these highly aerobatic maneuvers, we achieve, as expected, better performance by correctly leveraging the structure of the rotation group during the optimization routine.

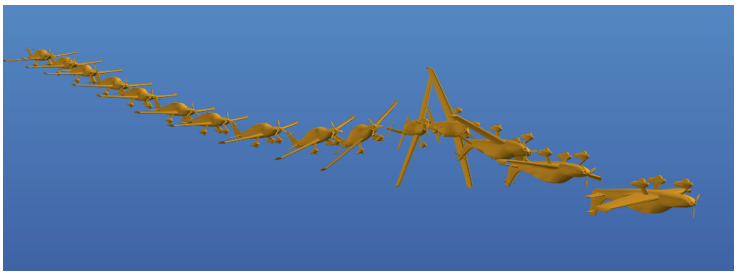


Fig. 4: Barrel roll trajectory computed by iterative MLQR using a terminal cost to encourage an upside-down attitude.

5.3 Tracking Control

This section provides examples using MLQR for tracking and stabilizing control for a quadrotor. We compare our MLQR controller to HFCA, a state-of-the-art geometric tracking controller [28] based on the Hopf Fibration.

MLQR Stabilization To test robustness and the domain of attraction of the MLQR controller, a quadrotor is linearized about hover at the origin and initialized from states uniformly sampled in $[-1, 1]$ for position, $[-5, 5]$ for linear and angular velocities, and uniformly over the entire space of rotations. The

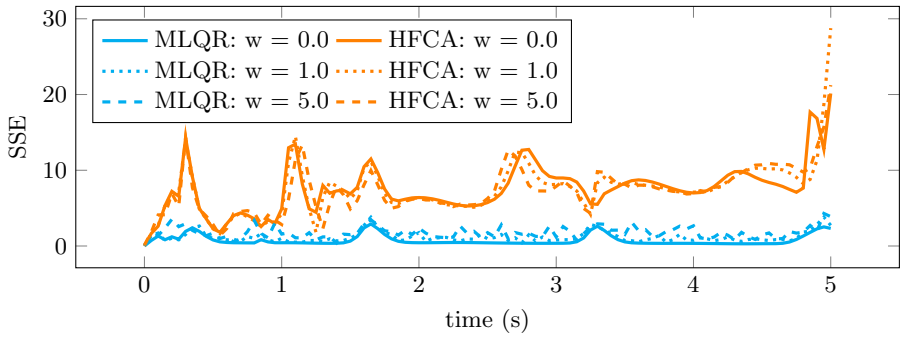


Fig. 5: Tracking error over time for the zig-zag trajectory using MLQR and the HFCA geometric controller. Band-limited white noise wrenches of variance w are added to the continuous dynamics.

success rates for 2000 trials are given in Table 1, where success is defined as the terminal error (calculated using (22)) being within a unit ball of radius 0.1 after 10.0 seconds. To further analyze the basins of attraction, we uniformly sampled over only roll (which goes singular at 90° for our Euler angle representation), and yaw. The results are shown in Fig. 6, where the plotted contours are the convex hull of the successfully stabilized initial conditions.

MLQR slightly exceeds the robustness of the HFCA geometric controller, while also being more generally applicable, easier to tune, and easier to implement. It is also noteworthy that naïvely using LQR with quaternions (incorrectly treating them as vectors) results in poorer performance than Euler angles, clearly demonstrating the value of the methodology presented in the current paper.

MLQR Tracking Time-varying MLQR is used to track the zig-zag trajectory generated in the previous section. To test tracking performance, zero-mean Gaussian noise was added as force and moments applied to the quadrotor.

The trajectories and tracking error for MLQR and HFCA are shown in Figs. 3a, and 5. While the MLQR controller dramatically out-performs HFCA, we found no significant difference in performance between MLQR and LQR using either roll-pitch-yaw Euler angles or a quaternion. Since the tracking controller only deals with relatively small state and attitude errors, this is unsurprising. Additionally, we found the MLQR controller ran about 20x faster than the HFCA controller.

6 Conclusions

We have presented a general, unified method for planning and control on rigid-body systems with arbitrary attitude using standard linear algebra and vector calculus. By applying these methods to LQR to correctly account for the group

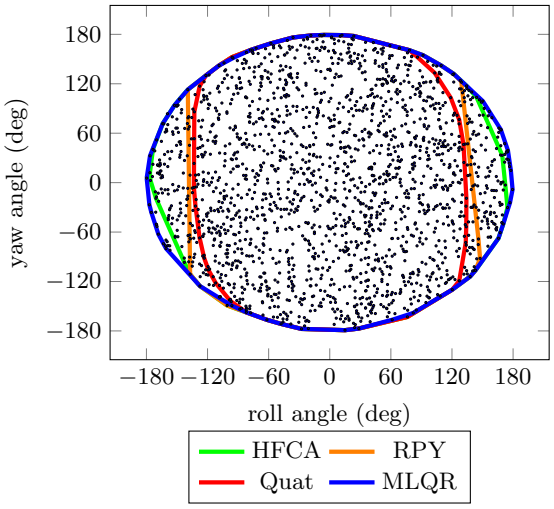


Fig. 6: Basins of attraction of 4 stabilizing controllers for yaw and roll, calculated from 2000 sample points (shown in black). All points within each contour were successfully stabilized by the controller.

Method	Success
Quat	61%
RPY	80%
HFCA	97%
MLQR	99%

Table 1: Success rate for Monte-Carlo analysis of quadrotor stabilization for 2000 initial conditions uniformly over position, orientation, and linear and angular velocities.

structure of rotations, we have matched or exceeded the performance of a state-of-the-art geometric controller designed specifically for quadrotors, while being more general, requiring less system-dependent tuning, having less computational overhead, and being easier to implement.

We have also demonstrated a straight-forward way to adapt nonlinear trajectory optimization techniques to work with quaternion-valued states. For these problems, we found the geodesic distance between quaternions (32) to be computationally efficient and provide excellent convergence in practice. We recommend the use of unit quaternions within planning and control algorithms for their simplicity, computational efficiency, and lack of singularities. We further recommend the use of Rodrigues parameters, or the Cayley map, as the quaternion error state.

Future areas of research include the application of these methodologies to more complex multi-body floating-base robots, such as humanoids and quadrupeds, as well as more in-depth analysis of the convergence behavior of the algorithms proposed in the current work.

Acknowledgements This material is based upon work supported by the National Science Foundation Graduate Research Fellowship Program under Grant No. DGE-1656518. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation

References

[1] Yaser Alothman and Dongbing Gu. "Quadrotor Transporting Cable-Suspended Load Using Iterative Linear Quadratic Regulator (iLQR) Optimal Control". In: *2016 8th Computer Science and Electronic Engineering (CEEC)*. Sept. 2016, pp. 168–173. DOI: 10.1109/CEEC.2016.7835908.

[2] Georges S. Aoude, Jonathan P. How, and Ian M. Garcia. "Two-Stage Path Planning Approach for Solving Multiple Spacecraft Reconfiguration Maneuvers". en. In: *The Journal of the Astronautical Sciences* 56.4 (Dec. 2008), pp. 515–544. ISSN: 0021-9142. DOI: 10.1007/BF03256564. URL: http://link.springer.com/10.1007/BF03256564 (visited on 01/08/2020).

[3] J. Baillieul. "Geometric Methods for Nonlinear Optimal Control Problems". en. In: *Journal of Optimization Theory and Applications* 25.4 (Aug. 1978), pp. 519–548. ISSN: 0022-3239, 1573-2878. DOI: 10.1007/BF00933518. URL: http://link.springer.com/10.1007/BF00933518 (visited on 01/07/2020).

[4] R. W. Brockett. "Lie Theory and Control Systems Defined on Spheres". en. In: *SIAM Journal on Applied Mathematics* 25.2 (Sept. 1973), pp. 213–225. ISSN: 0036-1399, 1095-712X. DOI: 10.1137/0125025. URL: http://epubs.siam.org/doi/10.1137/0125025 (visited on 01/07/2020).

[5] Nalin A. Chaturvedi, N. Harris McClamroch, and Dennis S. Bernstein. "Asymptotic Smooth Stabilization of the Inverted 3-D Pendulum". In: *IEEE Transactions on Automatic Control* 54.6 (June 2009), pp. 1204–1215. ISSN: 2334-3303. DOI: 10.1109/TAC.2009.2019792.

[6] Cedric de Crousaz et al. "Unified Motion Control for Dynamic Quadrotor Maneuvers Demonstrated on Slung Load and Rotor Failure Tasks". en. In: *2015 IEEE International Conference on Robotics and Automation (ICRA)*. Seattle, WA, USA: IEEE, May 2015, pp. 2223–2229. ISBN: 978-1-4799-6923-4. DOI: 10.1109/ICRA.2015.7139493. URL: http://ieeexplore.ieee.org/document/7139493/ (visited on 01/09/2020).

[7] Taosha Fan and Todd Murphey. "Online Feedback Control for Input-Saturated Robotic Systems on Lie Groups". en. In: *Robotics: Science and Systems XII* (2016). DOI: 10.15607/RSS.2016.XII.027. arXiv: 1709.00376. URL: http://arxiv.org/abs/1709.00376 (visited on 01/09/2020).

[8] Emil Fresk and George Nikolakopoulos. "Full Quaternion Based Attitude Control for a Quadrotor". In: *2013 European Control Conference (ECC)*. July 2013, pp. 3864–3869. DOI: 10.23919/ECC.2013.6669617.

[9] I. Garcia and J.P. How. "Trajectory Optimization for Satellite Reconfiguration Maneuvers with Position and Attitude Constraints". In: *Proceedings of the 2005, American Control Conference, 2005.* June 2005, 889–894 vol. 2. DOI: 10.1109/ACC.2005.1470072.

[10] Mathieu Geisert and Nicolas Mansard. "Trajectory Generation for Quadrotor Based Systems Using Numerical Optimal Control". In: *2016 IEEE International Conference on Robotics and Automation (ICRA)*. May 2016, pp. 2958–2964. DOI: 10.1109/ICRA.2016.7487460.

[11] Taylor A Howell, Brian E Jackson, and Zachary Manchester. "ALTRO: A fast solver for constrained trajectory optimization". In: *2019 IEEE International Conference on Intelligent Robots and Systems, IEEE*. 2019.

[12] Eric N. Johnson and Suresh K. Kannan. "Adaptive Trajectory Control for Autonomous Helicopters". en. In: *Journal of Guidance, Control, and Dynamics* 28.3 (May 2005), pp. 524–538. ISSN: 0731-5090, 1533-3884. DOI: 10.2514/1.6271. URL: https://arc.aiaa.org/doi/10.2514/1.6271 (visited on 01/07/2020).

[13] Marin Kobilarov. "Discrete Optimal Control on Lie Groups and Applications to Robotic Vehicles". In: *2014 IEEE International Conference on Robotics and Automation (ICRA)*. May 2014, pp. 5523–5529. DOI: 10.1109/ICRA.2014.6907671.

[14] Marin B. Kobilarov and Jerrold E. Marsden. "Discrete Geometric Optimal Control on Lie Groups". In: *IEEE Transactions on Robotics* 27.4 (Aug. 2011), pp. 641–655. ISSN: 1941-0468. DOI: 10.1109/TRO.2011.2139130.

[15] J.J. Kuffner. "Effective Sampling and Distance Metrics for 3D Rigid Body Path Planning". en. In: *IEEE International Conference on Robotics and Automation, 2004. Proceedings. ICRA '04. 2004.* New Orleans, LA, USA: IEEE, 2004, 3993–3998 Vol.4. ISBN: 978-0-7803-8232-9. DOI: 10.1109/ROBOT.2004.1308895. URL: http://ieeexplore.ieee.org/document/1308895/ (visited on 01/08/2020).

[16] T. Lee, M. Leok, and N. H. McClamroch. "Optimal Attitude Control of a Rigid Body Using Geometrically Exact Computations on SO(3)". en. In: *Journal of Dynamical and Control Systems* 14.4 (Oct. 2008), pp. 465–487. ISSN: 1079-2724, 1573-8698. DOI: 10.1007/s10883-008-9047-7. URL: http://link.springer.com/10.1007/s10883-008-9047-7 (visited on 12/19/2019).

[17] Taeyoung Lee, Melvin Leok, and N Harris McClamroch. "Geometric tracking control of a quadrotor UAV on SE (3)". In: *49th IEEE conference on decision and control (CDC)*. IEEE. 2010, pp. 5420–5425.

[18] Hao Liu, Xiafu Wang, and Yisheng Zhong. "Quaternion-Based Robust Attitude Control for Uncertain Robotic Quadrotors". In: *IEEE Transactions on Industrial Informatics* 11.2 (Apr. 2015), pp. 406–415. ISSN: 1941-0050. DOI: 10.1109/TII.2015.2397878.

[19] D. P. Mandic, C. Jahanchahi, and C. Cheong Took. "A Quaternion Gradient Operator and Its Applications". In: *IEEE Signal Processing Letters* 18.1 (Jan. 2011), pp. 47–50. ISSN: 1558-2361. DOI: 10.1109/LSP.2010.2091126.

[20] F Landis Markley and John L Crassidis. *Fundamentals of spacecraft attitude determination and control*. Vol. 33. Springer, 2014.

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

[22] Nathan Michael et al. "The GRASP Multiple Micro-UAV Testbed". In: *IEEE Robotics & Automation Magazine* 17.3 (2010), pp. 56–65.

[23] Alessandro Saccon, John Hauser, and A. Pedro Aguiar. "Optimal Control on Lie Groups: The Projection Operator Approach". In: *IEEE Transactions on Automatic Control* 58.9 (Sept. 2013), pp. 2230–2245. ISSN: 2334-3303. DOI: 10.1109/TAC.2013.2258817.

[24] Joan Solà. "Quaternion Kinematics for the Error-State Kalman Filter". en. In: *arXiv:1711.02508 [cs]* (Nov. 2017). arXiv: 1711.02508 [cs]. URL: http://arxiv.org/abs/1711.02508 (visited on 09/03/2019).

[25] Karlheinz Spindler. "Optimal Control on Lie Groups with Applications to Attitude Control". en. In: *Mathematics of Control, Signals, and Systems* 11.3 (Sept. 1998), pp. 197–219. ISSN: 0932-4194, 1435-568X. DOI: 10.1007/BF02741891. URL: http://link.springer.com/10.1007/BF02741891 (visited on 01/07/2020).

[26] John Stuelpnagel. "On the parametrization of the three-dimensional rotation group". In: *SIAM review* 6.4 (1964), pp. 422–430.

[27] George Terzakis, Manolis Lourakis, and Djamel Ait-Boudaoud. "Modified Rodrigues Parameters: An Efficient Representation of Orientation in 3D Vision and Graphics". en. In: *Journal of Mathematical Imaging and Vision* 60.3 (Mar. 2018), pp. 422–442. ISSN: 0924-9907, 1573-7683. (Visited on 12/19/2019).

[28] Michael Watterson and Vijay Kumar. "Control of quadrotors using the HOPF fibration on SO (3)". In: *Robotics Research*. Springer, 2020, pp. 199–215.

[29] Michael Watterson et al. "Trajectory Optimization On Manifolds with Applications to $SO(3)$ and $\mathbb{R}^3 \times S^2$." In: *Robotics: Science and Systems*. 2018.

[30] Bong Wie and Peter M Barba. "Quaternion feedback for spacecraft large angle maneuvers". In: *Journal of Guidance, Control, and Dynamics* 8.3 (1985), pp. 360–365.

[31] Grady Williams, Andrew Aldrich, and Evangelos A. Theodorou. "Model Predictive Path Integral Control: From Theory to Parallel Computation". en. In: *Journal of Guidance, Control, and Dynamics* 40.2 (Feb. 2017), pp. 344–357. ISSN: 0731-5090, 1533-3884. DOI: 10.2514/1.G001921. URL: http://arc.aiaa.org/doi/10.2514/1.G001921 (visited on 01/09/2020).

[32] Dongpo Xu, Yili Xia, and Danilo P. Mandic. "Optimization in Quaternion Dynamic Systems: Gradient, Hessian, and Learning Algorithms". In: *IEEE Transactions on Neural Networks and Learning Systems* 27.2 (Feb. 2016), pp. 249–261. ISSN: 2162-2388. DOI: 10.1109/TNNLS.2015.2440473.

[33] M. Zefran, V. Kumar, and C.B. Croke. "On the Generation of Smooth Three-Dimensional Rigid Body Motions". In: *IEEE Transactions on Robotics and Automation* 14.4 (Aug. 1998), pp. 576–589. ISSN: 2374-958X. DOI: 10.1109/70.704225.

[34] Miloš Žefran, Vijay Kumar, and Christopher Croke. "Metrics and Connections for Rigid-Body Kinematics". en. In: *The International Journal of Robotics Research* 18.2 (Feb. 1999), pp. 242–1. ISSN: 0278-3649.