Development of an LQR Controller for a Jet Vanes Thrust-Vectored Rocket

Patrick T. Barry*, Albert Zheng†, Karsten Caillet‡, David Reynolds§
Georgia Institute of Technology, Atlanta, GA, 30332

This paper outlines the task of developing a control system for a rocket controlled by jet vanes, small diamond-shaped actuators placed in exhaust flow of a motor to manipulate the rocket's attitude along its roll and yaw axes. With usage of a custom aerodynamic model, an in-house 6-DOF simulation was written to analyze the rocket trajectory. Based on the rocket's flight profile, several trim points were established to linearize rocket dynamics for use of a Linear Quadratic Regulator (LQR) controller. Using the rocket's state in flight, the controller selects a gain matrix and computes optimal roll and yaw moments to achieve a specified desired attitude. The desired attitude is comprised of a zero roll angle and a variable setpoint yaw angle. A Monte Carlo analysis was performed to ensure the controller met its goals under various rocket parameters and flight conditions. The result is a simulation-verified control system architecture as well as a step-by-step process to developing LQR controllers for any rocket and actuator configuration.

I. Introduction

The Guidance, Navigation, and Control (GNC) project within the Ramblin' Rocket Club at the Georgia Institute of Technology is a team of students that design and build actively controlled L1- and L3-class rockets. Most recently, the focus of the GNC project has been to become the first collegiate rocketry team to fly a jet vanes rocket (JVR), a vehicle steered by deflecting small diamond-shaped vanes in the flow exiting from the nozzle of the solid propellant motor of the rocket. The goal of JVR is to achieve active attitude control along the roll and yaw axes of the rocket over the course of its powered ascent. The combination of jet vanes being a novel technology within collegiate rocketry and complex vehicle dynamics resulted in the need for high-fidelity simulations and the development of a robust control system. After establishing a 6-DOF (degree of freedom) MATLAB simulation, an LQR controller was integrated to guide the JVR to desired attitude orientations over the course of its powered ascent. Because LQR control demands a linear, time-invariant system, JVR dynamics were linearized at several points in the flight profile. The process of writing the simulations and developing the LQR controller is detailed as a basis for developing an optimal control system architecture for any given launch vehicle.

II. Dynamical Modeling

The first step in the development of a real time optimal controller is defining a mathematical model that closely resembles true system dynamics. The dynamic model serves three main purposes: it validates passive fin stability, is used for controller linearization and gain selection, and generates a reference trajectory to validate the team's in-flight navigation algorithm, which is not detailed in this paper. As with most real-world systems, the 6-DOF dynamics of a rocket are highly stochastic, nonlinear, and time variant. However, many valid simplifications can be made to avoid severe reduction in real-time controller performance. Since the active control time is less than the short motor burn time, and the altitude doesn't exceed 10,000 feet, it is assumed that Earth's rotation and differential changes in gravity are negligible. A custom inertial flat Earth frame $i = \{x^i, y^i, z^i\}$ is defined as having origin at the launch rail, with the x-axis pointing straight up, y-axis pointed in the downrange direction, and the z-axis in the direction that satisfies the right-hand rule. Figure 1 visualizes sign and frame conventions on the rocket's body frame $b = \{x^b, y^b, z^b\}$.

*GNC Avionics & Controls Lead, Undergraduate, Daniel Guggenheim School of Aerospace Engineering, AIAA Student Member 1418912
†GNC Controls Team Member, Undergraduate, Daniel Guggenheim School of Aerospace Engineering, AIAA Student Member 1603522
‡GNC Controls Team Member, Undergraduate, Daniel Guggenheim School of Aerospace Engineering, AIAA Student Member 1537725
§GNC Controls Team Member, Undergraduate, Daniel Guggenheim School of Aerospace Engineering, AIAA Student Member 1789019

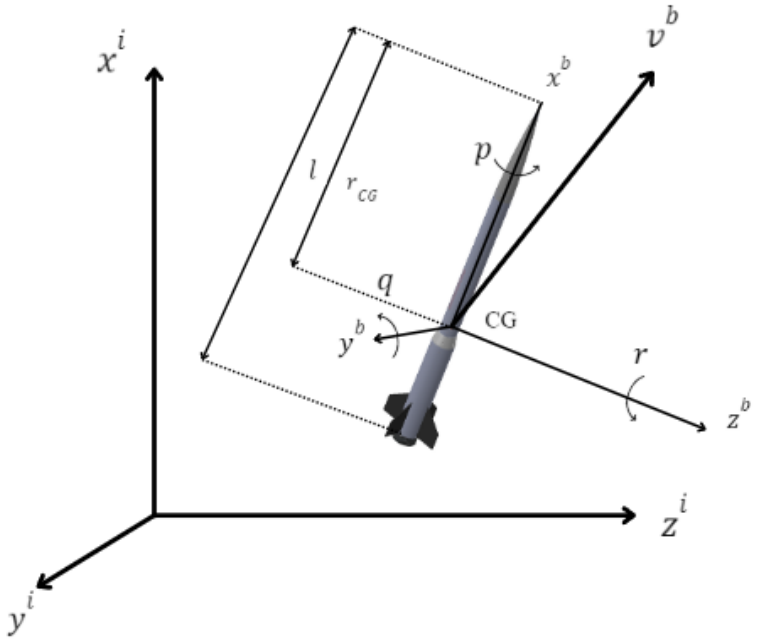

Figure 1. Sign and frame conventions for the JVR.

A. Rocket Equations of Motion

The mathematical equations of motion describe the 6-DOF dynamics of a 13-element state vector for position, body velocity, attitude, and angular velocity of the rocket, as shown in Equations 1a-1d.

$$\dot{\vec{p}}^i = \vec{q}^*_{bi} \otimes \vec{v}^b \otimes \vec{q}_{bi} \tag{1a}$$

$$\dot{\vec{v}}^b = \frac{\vec{F}_b}{m} - \vec{\omega}^b_{bi} \times \vec{v}^b \tag{1b}$$

$$\dot{\vec{q}}_{bi} = \frac{1}{2} \begin{bmatrix} -pq_x - qq_y - rq_z \\ pq_w + rq_y - qq_z \\ qq_w - rq_x + pq_z \\ rq_w + qq_x - pq_y \end{bmatrix} \tag{1c}$$

$$\dot{\vec{\omega}}^b_{bi} = I_b^{-1}(M^b - \vec{\omega}^b_{bi} \times I_b \vec{\omega}^b_{bi}) \tag{1d}$$

The gravity force in the body frame is computed by first defining gravity in the flat Earth frame as $\vec{F}^i_G = \begin{bmatrix} -g & 0 & 0 \end{bmatrix}^T$. Then, gravity is rotated to the body frame using the quaternion state: $\vec{F}^b_G = \vec{q}_{bi} \otimes \vec{F}^i_G \otimes \vec{q}^*_{bi}$, where $\otimes$ is the quaternion multiplication operator. The thrust force in the body frame is computed by assuming the projected motor thrust curve only has a component in the x-direction.

The aerodynamic forces and moments in the body frame are based on modeling by a separate computational fluid dynamics (CFD) team. The team performed extensive CFD analysis on the rocket aerodynamics at flight speeds ranging from Mach 0.1 to 0.7 at angles of attack ranging from 0° to 5°. Simulation data then underwent a least-squares fit to produce equations describing the aerodynamics of the JVR. This CFD analysis assumes a planar vehicle in which angle of attack only occurs due to yawing. The analysis also accounts for damping moment and roll damping effects. The provided aerodynamic moments are about the initial center of gravity at t = 0, so a moment transfer is needed at every

time step to compute the aerodynamic moment about the current center of gravity, as the CG changes with time.

Since mass and mass moments of inertia are time-variant, wet and dry values are saved and linearly interpolated at each time step to compute the current mass and inertia values, given the current time of flight. The wet and dry values are obtained from CAD models developed by a separate structures team.

B. Open Loop Simulation Architecture

Validation of the dynamic model is then performed in an open-loop JVR simulation. The equations of motion are represented as a system of 13 first-order differential equations in terms of $\dot{\vec{x}} = f(\vec{x}, t)$. This representation computes corresponding state derivatives that predict the future state one time step later. Open loop rocket stability is verified by observing that angular velocity and angle of attack both converge to zero within a few seconds.

III. Controller Development

A. Linear Quadratic Regulator for Rocket Control

Linear Quadratic Regulator (LQR) is a form of optimal control for linear systems that is favorable due to its ability to handle multi-input, multi-output systems and simple tuning process. In its most basic form, the LQR problem can be stated as follows: Given the linear dynamics

$$\dot{\vec{x}} = A\vec{x} + B\vec{u} \tag{2}$$

find a gain matrix K that follows the control feedback law

$$\vec{u} = -K\vec{x} \tag{3}$$

such that a cost function J is minimized, with J defined as

$$J = \int_0^\infty (\vec{x}^\intercal Q\vec{x} + \vec{u}^\intercal R\vec{u})\, dt \tag{4}$$

with Q defined as a positive-semidefinite symmetric matrix and R defined as a positive-definite symmetric matrix.[1] The utility of an LQR controller stems from the Q and R matrices; the Q matrix scales the cost according to the states, so increasing values within Q can be viewed as prioritizing system performance. Increasing values in the R matrix increases the cost to actuate, making the system more cautious to exert effort to reach a desired state. For any given set of A, B, Q and R matrices, an optimal gain matrix K can be derived to minimize the cost function J by solving an equality known as the Algebraic Riccati equation; assuming a continuous-time system, this derivation of K can be abstracted away in MATLAB using $K = lqr(A, B, Q, R)$.

The control goal of the JVR is to pursue actively stabilized vertical flight before turning and holding angled flight, with the rocket returning to vertical flight at the end of its powered ascent. The motor used for the JVR is the commercially available Aerotech N1100 motor, which contains a burn time of 13.1 seconds; the JVR can only perform controlled maneuvers during the motor burn time, leading to the following flight profile:

1) After motor ignition, let the rocket passively stabilize for one second.
2) From one second to four seconds after motor ignition, utilize the jet vanes to actively stabilize the rocket in vertical flight.
3) Beginning at four seconds, and until nine seconds after ignition, execute a yaw maneuver to turn and hold the jet vanes rocket at a five-degree angle with respect to the vertical.
4) At nine seconds after ignition and until burnout, turn the rocket back to vertical flight and maintain vertical flight until burnout.

At all times in the flight profile, it is desired to maintain the vehicle at zero roll. While it is possible to control all three modes of rotation (roll, pitch, and yaw) with jet vanes, only two of the three modes can be controlled simultaneously. This is because the jet vanes work by producing couples on the vehicle's airframe via deflection of exhaust from the motor's nozzle. The sideforce produced by each jet vane can be converted into a pure moment on the vehicle by ensuring that vanes across from each other actuate to control the same mode of rotation.

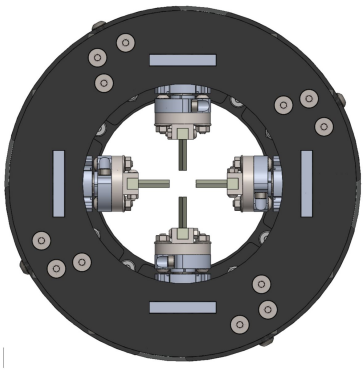

Figure 2. Jet vanes assembly. Pairs of vanes act to produce couple moments on the rocket airframe

To simplify the controls scheme, two of the vanes are allocated to roll, and two are dedicated to yaw. This removes the vehicle's ability to control pitch, but because the JVR is passively stable with its fins, it is reasonable to assume that the rocket will not meaningfully pitch over the duration of its powered flight. Additionally, it is assumed the air density remains constant over the course of the flight, as the altitude of the JVR is expected to change by only 3000 feet during the course of its powered ascent; this allows the position of the rocket to be neglected in the state vector, yielding the following set of states:

$$\vec{x} = [u \;\; v \;\; w \;\; p \;\; q \;\; r \;\; q_0 \;\; q_1 \;\; q_2 \;\; q_3]^\intercal \tag{5}$$

Because jet vanes work by creating moments on the rocket's airframe, the control vector $\vec{u}$ is composed of control input moments. In a practical implementation, turning the jet vanes requires a servo deflection angle, which can be derived from an optimal control moment using moment arm lengths, a sideforce to vane angle relationship, and a servo-to-vane gear ratio. Finally, to establish the rocket dynamics in a linearized form, because the thrust acts as an external input to the system, the thrust is included as an element in the control input vector.

B. Trim Point Formulation

As seen in section II, the rocket's dynamics are highly nonlinear; additionally, with a non-constant thrust value and shifting center of gravity, these dynamics are also time-variant. LQR control is a form of control for linear, time-invariant systems, so there must be a way to convert the complex rocket dynamics into this form. This is done by establishing trim points: regions in the state-space over which the rocket can be considered as a linear, time-invariant system. With enough trim points, the entire flight profile of the JVR can be covered, with an optimal controller derived for each trim point. A trim point is formally defined as some set of values for $\dot{\vec{x}}$, $\vec{x}$, and $\vec{u}$ that are dynamically feasible, i.e., they satisfy the nonlinear dynamics $\dot{\vec{x}} = f(\vec{x}, \vec{u})$.

The set of trim points were established keeping in mind several aspects of the rocket's dynamics that could be changing: motor thrust, CG location, moments of inertia, rocket airspeed, and angle of the rocket with respect to the vertical. A nominal flight profile of the jet vanes rocket was established using a basic OpenRocket simulation. This provided expected values of the rocket's parameters, namely airspeed and thrust, at discrete intervals over the course of its flight. A baseline trim point for each second of flight could then be established. Noting that the OpenRocket simulation reports a primarily vertical flight, the reported airspeed of the rocket acts effectively as u, the rocket's velocity along its body x-axis. Then, v and w are assumed to be zero at every trim point, meaning the rocket has no angle of attack in any trim point. The rotation rates of the rocket (p, q, r) are also zero at every trim point, and for vertical flight, the quaternion is $q = [1, 0, 0, 0]$. Additionally, no trim points included the rocket actively turning, so the control input moments were taken to be zero for each trim point, and the thrust reported by OpenRocket at each point in time was directly included.

Variations can then be added to each baseline trim point at each second of flight to ensure that the flight profile is fully encapsulated by the set of trim points. For each baseline trim point, the airspeed was varied in intervals such that the drag force magnitude changed by an arbitrary 8%. This was performed at five intervals above and below the airspeed reported by OpenRocket, resulting in eleven trim points for each second of flight. Finally, to take into account the fact that the rocket would begin turning to a 5° angle beginning in the fourth second of flight, trim

points were added for all seconds after the beginning of this turning maneuver to reflect an angled flight, changing the quaternion value to that of a rocket that has yawed 5°, $q = [0.9990, 0, 0, 0.0436]$. An optimization algorithm was then utilized that constrained the rates of change of both the rotational velocities (p, q, r) and the quaternion to zero to find a dynamically feasible combination of $\dot{\vec{x}}$, $\vec{x}$, and $\vec{u}$ that comprises a trim point. Thus, each trim point considers time-varying quantities, uncertainty in the rocket's airspeed during the flight, and the yaw maneuver executed by the JVR, as well as satisfying the nonlinear equations of motion; in total, there are 429 trim points for the 13-second powered ascent.

Trim points were generated in an automated fashion using a script that considers the data provided by OpenRocket and uses the dynamic model of the rocket to appropriately space the airspeeds for the trim points at each second of flight. This greatly accelerates the pace of developing a controller as minor changes were made to the rocket's aerodynamic model and control goals during development.

C. Linearization of Equations of Motion

With the set of 429 trim points covering the rocket's flight profile during its powered ascent, the next step to obtaining an in-flight LQR controller was to linearize the dynamical equations of motion about each trim point such that the motion of the rocket could be described by Equation 2. Let $\dot{\vec{x}} = f(\vec{x}, \vec{u})$ represent the system of equation that is the dynamic model discussed in Section II. This system of equations can be rearranged and rewritten as

$$F(\dot{\vec{x}}, \vec{x}, \vec{u}) = F(\vec{\eta}) = \vec{0} \tag{6}$$

Here, $\vec{\eta}$ is a vector that concatenates the $\dot{\vec{x}}$, $\vec{x}$, and $\vec{u}$ vectors for mathematical simplicity. Letting $\vec{\eta_0}$ represent a trim point, a first-order Taylor expansion can be performed about that trim point, yielding

$$F(\vec{\eta}) = F(\vec{\eta_0}) + \frac{\partial F(\vec{\eta_0})}{\partial \vec{\eta}}(\vec{\eta} - \vec{\eta_0}) \tag{7}$$

By definition, $F(\vec{\eta_0}) = 0$, as each trim point satisfies the nonlinear equations of motion. Expanding the partial derivative term, and noting that the nonlinear system of equations is equal to the zero vector,

$$\frac{\partial F(\vec{\eta_0})}{\partial \vec{\eta}}(\vec{\eta} - \vec{\eta_0}) = \frac{\partial F(\dot{\vec{x_0}})}{\partial \dot{\vec{x}}}(\dot{\vec{x}} - \dot{\vec{x_0}}) + \frac{\partial F(\vec{x_0})}{\partial \vec{x}}(\vec{x} - \vec{x_0}) + \frac{\partial F(\vec{u_0})}{\partial \vec{u}}(\vec{u} - \vec{u_0}) \approx \vec{0} \tag{8}$$

The partial derivative of a system of equations with respect to a vector simply yields a matrix, so the above can be rewritten as

$$E\Delta\dot{\vec{x}} + A'\Delta\vec{x} + B'\Delta\vec{u} \tag{9}$$

where a Δ term represents the difference from the trim point. This equation can be reshaped into the form of Equation 2, resulting in

$$\Delta\dot{\vec{x}} = -E^{-1}A'\Delta\vec{x} - E^{-1}B'\Delta\vec{u} \tag{10}$$

$$\Delta\dot{\vec{x}} = A\Delta\vec{x} + B\Delta\vec{u} \tag{11}$$

Thus, Taylor expanding about each trim point allows one to obtain linearized dynamics in terms of deviations from that trim point.[2]

This approach requires generation of a different A and B matrix for every trim point, which can be efficently computed using MATLAB's Symbolic Math Toolbox. The nonlinear system of equations representing the JVR's dynamics was written in terms of symbolic variables, with the E, A', and B' matrices computed in terms of variables instead of numbers with usage of the built-in $jacobian()$ function. Once a symbolic form for A and B had been derived, all rocket parameters and trim point values are plugged in to obtain numerical A and B matrices for every trim point. Thrust, rocket mass, CG location, and moments of inertia were all linearly interpolated based on which second of flight a trim point came from. At this point, when the rocket's state closely matched that of a trim point, linearized dynamics could be used to describe the rocket's motion in the neighborhood of that trim point.

D. Gain Matrix Generation

The usefulness of LQR lies in its tunability through the Q and R matrices. The cost function found in Equation 4 produces a quadratic cost formulation, with scaling Q associated with valuing system performance more, and scaling R making actuation more expensive for the system. For this project, Q and R were made to be diagonal matrices, meaning that the first diagonal entry in Q produces a cost on the first element in the state vector $\vec{x}$, the second diagonal entry scales cost for the second element of $\vec{x}$, and so on.

At this point, it is necessary to discuss the metric of controllability as it relates to LQR. A gain matrix can only be generated for a given combination of A, B, Q, and R if the dynamical system is controllable. A system is defined as controllable if its controllability matrix, C, has a rank equal to the number of states in the system[1]. C is defined as

$$C = [B \;\; AB \;\; \cdots \;\; A^{n-1}B] \tag{12}$$

Two aspects of the JVR harm its controllability. Notably, the first element of the quaternion, q_0, is uncontrollable, as it represents more of a normalization factor than an attitude; thrust, roll moment, and yaw moment cannot change value of q_0 directly. Additionally, as previously discussed, only roll moment and yaw moment are controlled by the JVR, yet the angular rates and quaternion contain elements that relate to pitch. Thus, a fake pitching moment control input was added to the control input vector $\vec{u}$, and q_0 was removed from the state vector $\vec{x}$. This yields a nine-element state vector: velocity and angular rates along the three body axes and three quaternion elements. Additionally, the control input vector becomes four elements: roll, pitch, and yaw moments, as well as thrust, which is included due to its status as an external input to the system, as previously discussed.

Leveraging the ability of LQR to tune the cost for each individual element of the state and control input vectors, diagonal values within the Q and R matrices were strategically selected to reflect the physical rocket's ability to steer itself. Because the JVR's control goals were concerned with its motion about the roll and yaw axes, the diagonal elements of Q associated with roll rate (p), yaw rate (r), roll attitude (q_1), and yaw attitude (q_3) were set at 1000, with all other diagonal elements set to 0.01. Similarly, because roll and yaw moments were physically modifiable by the jet vanes assembly, their associated elements in the R matrix were set to 0.01, denoting low cost to actuate about these axes. All other diagonal elements in the R matrix were set to 1000, since thrust and pitching moment cannot be controlled by the actual system. This LQR formulation is summarized below:

$$\vec{x} = [u \;\; v \;\; w \;\; p \;\; q \;\; r \;\; q_1 \;\; q_2 \;\; q_3]^\intercal \tag{13}$$

$$\vec{u} = [M_{roll} \;\; M_{pitch} \;\; M_{yaw} \;\; T]^\intercal \tag{14}$$

$$Q = diag([0.01, 0.01, 0.01, 1000, 0.01, 1000, 1000, 0.01, 1000]) \tag{15}$$

$$R = diag([0.01, 1000, 0.01, 1000]) \tag{16}$$

With this Q and R used over the full flight profile, every trim point now contained the matrices A, B, Q, and R. Thus, for each trim point, an optimal gain matrix K could be derived using MATLAB's $lqr()$ function.[3] All trim points were then saved in a MATLAB struct array, with each struct representing a trim point containing the second of flight, $\vec{x_0}, \vec{u_0}$, A, B, Q, R, and K.

The purpose of the trim points is to ensure the gain matrix used for any set of controls accurately reflects the rocket's dynamics at any given point during the powered ascent. As a result, it becomes necessary to select a trim point that most closely matches the current rocket's state before computing the controls using a given trim point's gain matrix K.[4] This task is performed by the "LQR Gain Selector" algorithm. Given the rocket's current state and the time since launch, the set of trim points is looped through to check which most closely matches the current state. First, only trim points made for the current second of flight that the rocket is in are compared; next, the distance in the state-space from each trim point is computed. The trim point that the current rocket state contains the minimum distance from is chosen as the current trim point, and its gain matrix K is selected as the optimal gain matrix for the next control action.

Once an optimal gain matrix is found for the current state, the control input at the current time step is computed based on the current state's deviation from a desired reference state. The reference state is a state vector with the same elements as the state vector used for the controller, from Equation 13; the reference velocity components (u, v, and w) match those of the trim point's, the reference angular velocity (p, q, and r) components are all zero, and the reference quaternion depends on what the current time of flight is. Based on the control goals of the rocket discussed in section

Algorithm 1 LQR Gain Selector

Given: $t_{current} \in [0, t_{burnout}]$; $\vec{x}_{current}$; $[t, \vec{x}, K] \in TrimPoints$
$t \leftarrow round(t_{current})$
$d_{min} \leftarrow 1e9$
for p in TrimPoints **do**
  if $p.t \neq t$ **then**
    continue
  end if
  $d \leftarrow (\vec{x}_{current} - p.\vec{x})^T(\vec{x}_{current} - p.\vec{x})$
  if $d < d_{min}$ **then**
    $d_{min} \leftarrow d$
    $p_{closest} \leftarrow p$
  end if
end for
$K \leftarrow p_{closest}.K$

III.A, the reference quaternion is either a quaternion representing a straight, vertical vehicle or a vehicle that has yawed 5°. With a gain matrix and a reference state determined, the controls are computed as

$$\vec{u} = -K(\vec{x}_{current} - \vec{x}_{reference}) \tag{17}$$

Because only roll and yaw can be controlled by the jet vanes, only the elements of the control input vector $\vec{u}$ that represent the optimal roll and yaw moments are used. The above formulation fully describes how an optimal control moment is found at any time during the rocket's powered ascent.

E. LQR Controller Validation

The first step to LQR controller validation is to check if the linearized A and B matrices about a given trim point closely resemble the nonlinear dynamics operating near that same trim point. This check takes the form of comparing the solved trajectory of the linearized system to the solved trajectory of the nonlinear system, both using MATLAB's $ode45()$. This was performed and the system's state closely aligned under gain-scheduled linearized dynamics as compared with the trajectory utiliing nonlinear equations of motion.

Next, Monte Carlo simulations are run with the closed-loop controller to ensure the rocket's yaw and roll angles closely matches the reference desired attitude. Note that the linearized model is not used to perform closed-loop simulations and only exists to provide verification that it matches the nonlinear model accurately. A Monte Carlo simulation is a manner of proving controller stability and performance over a wide variety of rocket parameters and flight conditions. Several simulation variables are varied over the course of many flight simulations; these are called Monte Carlo parameters. In this case, rocket mass and moments of inertia, center of gravity location, motor burn time, thrust values, and wind speeds were all varied. Each parameters' variation was represented as a Gaussian random variable, with a 3σ distribution equal to 10% of each parameter's mean value. [5] Figures 3 and 4 show the 3D trajectory and yaw angle of the simulated JVR over a set of 1000 Monte Carlo samples.

Figure 5 shows the mean, standard deviation, and 3-sigma deviation of the yaw angle results alongside the mission requirements of the JVR. The JVR's yaw angle successfully fits within requirements for almost all the samples, which validates the LQR controller performance in simulations, confirming readiness for a real flight.

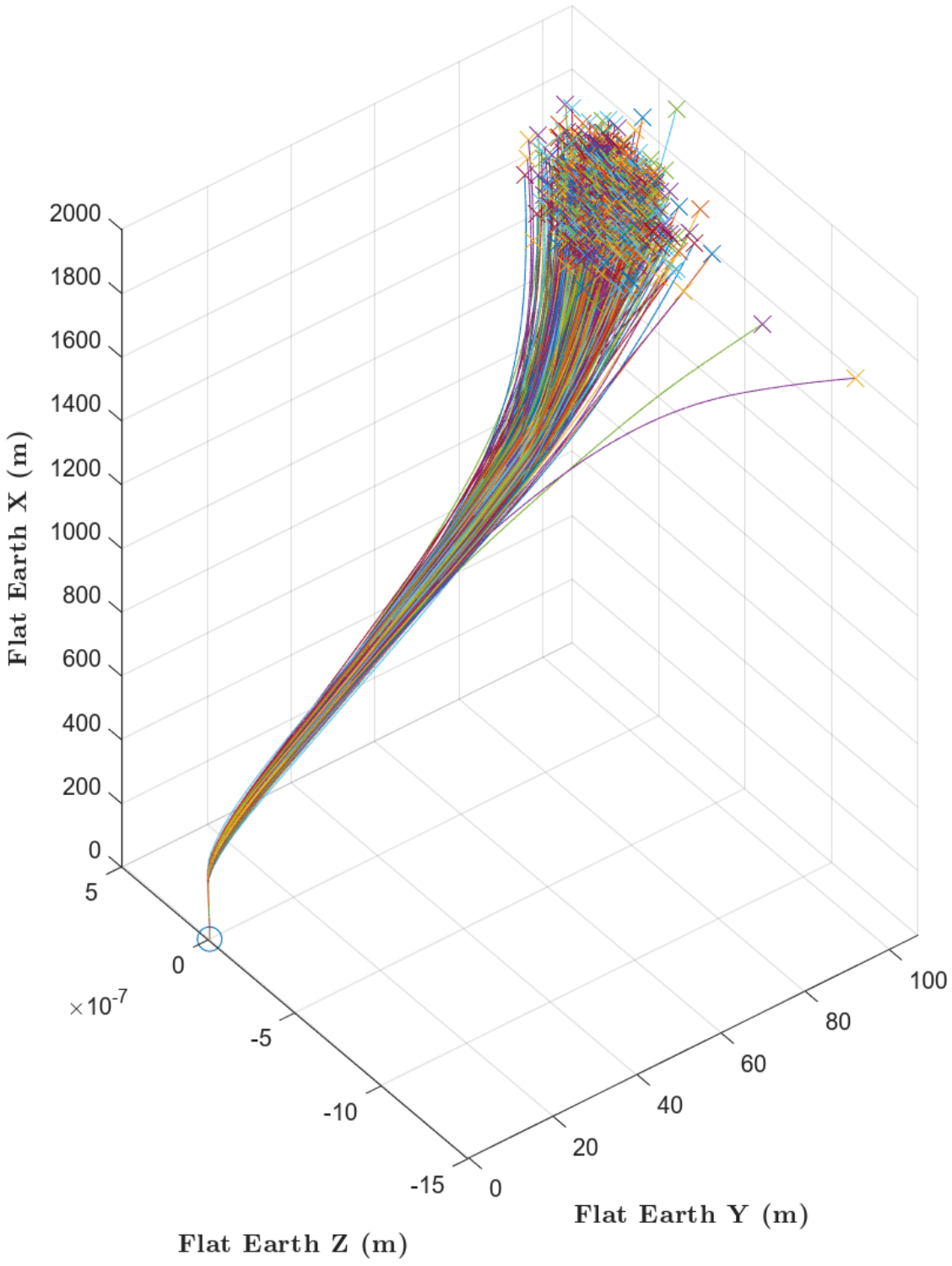

Figure 3. Closed-loop 3-dimensional trajectories for a 1000-sample Monte Carlo simulation.

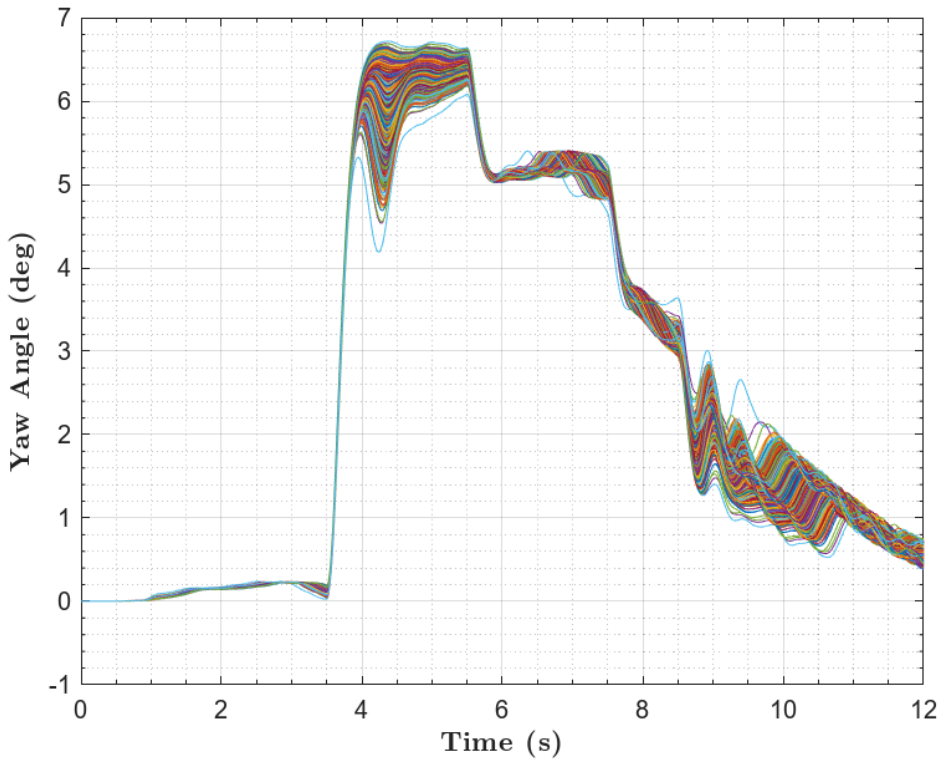

Figure 4. Yaw angles for 1000-sample Monte Carlo simulation.

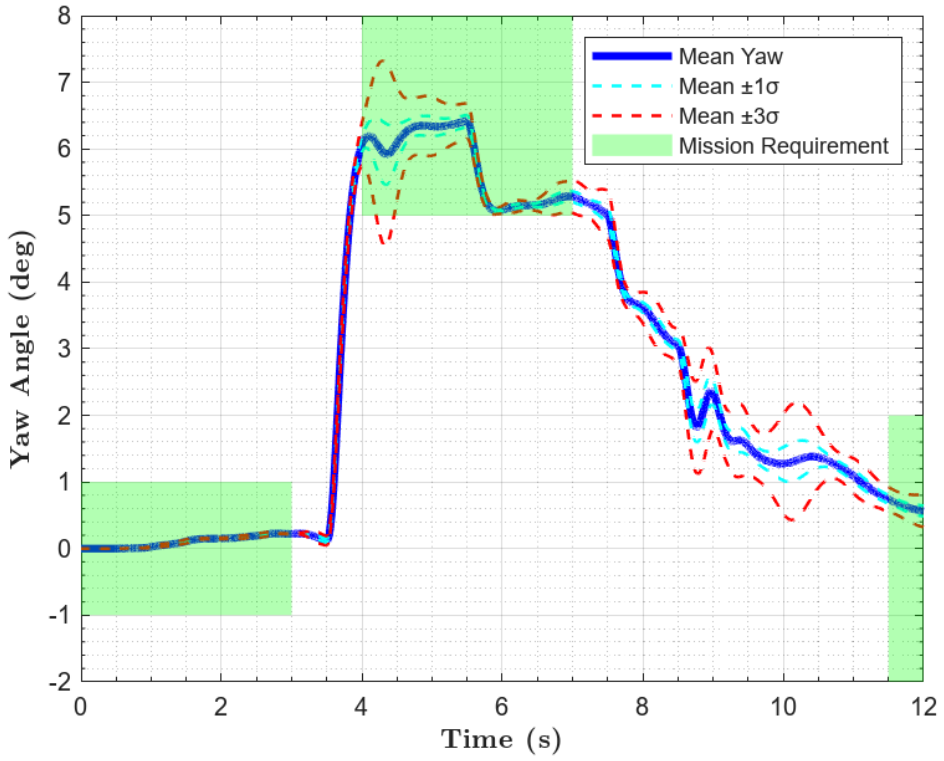

Figure 5. Yaw angle mean, standard deviation, and 3σ distribution with mission requirements for 1000-sample Monte Carlo simulation.

IV. Conclusion

The work presented in this paper represents the set of steps necessary in implementing a Linear Quadratic Regulator controller for a 6-DOF rocket thrust-vectored by jet vanes. While the final controller for the vehicle is tuned for specific structural and dynamical parameters, the architecture around this controller may apply to any nonlinear dynamical system. An open-loop simulation may be built around the foundation of the rocket's dynamics before linearizing distinct points in the flight profile. Each of these trim points in the state-space may then be leveraged to derive an optimal controller for usage when the rocket's state closely matches that of the trim point. A Monte Carlo analysis subsequently proves controller performance in the absence of certainty in system parameters. The advantage of linear controllers lies in their algorithmic simplicity, so laying the groundwork for their implementation on a vehicle with nonlinear dynamics greatly reduces the controller development time for a complex system.

References

[1] Ogata, K., *Modern Control Engineering, Fifth Edition*, Pearson, 25 August 2009.

[2] Kisabo, A. B., and Adebimpe, A. F., "State-Space Modeling of a Rocket for Optimal Control Design," *InTech Open*, 5 June 2019.

[3] Sopegno, L., Livreri, P., Stefanovic, M., and Valavanis, K. P., "Linear Quadratic Regulator: A Simple Thrust Vector Control System for Rockets," *30th Mediterranean Conference on Control and Automation*, June 2022.

[4] Kolosa, D., "Implementing a Linear Quadratic Spacecraft Attitude Control System," *ScholarWorks at WMU*, December 2015.

[5] Hanson, J., and Beard, B., "Applying Monte Carlo Simulation to Launch Vehicle Design and Requirements Analysis," *NASA Technical Report*, 1 September 2010.