# AE353 Project 1: Gimbal/Rotor Simulation for State Feedback Controller

Austin A. Zickur
*University of Illinois Urbana-Champaign, Urbana, Illinois, 61820*

**For Project 1, controllability is the central trait that is tested by using linearization at an equilibrium point to find a constant control gain-matrix, K, that will be used in a controller for the original, nonlinearized equations of motion. The system's movement is further examined by changing values and using the controller.**

## I. Nomenclature

| | | |
|---|---|---|
| $q1$ | = | angle of platform (rad) |
| $q2$ | = | angle of gimble (rad) |
| $v1$ | = | angular velocity of platform (rad/s) |
| $v2$ | = | angular velocity of gimble (rad/s) |
| $v3$ | = | angular velocity of rotor (rad/s) |
| $tau2$ | = | torque applied by platform to gimble (N(m)) |
| $tau3$ | = | torque applied by gimble to rotor (N(m)) |
| $A$ | = | state coefficient matrix |
| $B$ | = | input coefficient matrix |
| $Amatrix$ | = | matrix A for linearized model |
| $Bmatrix$ | = | matrix B for linearized model |
| $nonlinA$ | = | matrix A for nonlinearized model |
| $nonlinB$ | = | matrix B for nonlinearized model |
| $eigen$ | = | eigenvalues for controller (A-BK) |
| $f$ | = | vector-valued function for linearized model |
| $h$ | = | vector-valued function for nonlinearized model |
| $q_e$ | = | equilibrium angle values |
| $v_e$ | = | equilibrium angular velocity values |
| $tau_e$ | = | equilibrium torque values |
| $x$ | = | state variable matrix |
| $u$ | = | input variable matrix |
| $K$ | = | constant control gain-matrix |

## II. Introduction

This design project involves linearizing a system involving equations of motion in Ordinary Differential form by taking an equilibrium angle value for the gimble and rotor, and considering initial angular velocity of the rotor (100 revolutions/minute). After linearizing, we find a matrix K for a controller and set negative eigenvalues for the matrix (A-BK). Then, using the determined value for K in (nonlinA-nonlinB K), we can test the controller's functionality on the nonlinearized model in moving the gimbal and rotor to the set equilibrium points. The controller is successful if angular velocity and torques reach 0 after some finite time and angle reaches equilibrium angle after some finite time. Furthermore, the system's nature can be examined by noting the changes that occur after altering angles, eigenvalues, and graphing different models with run times related to eigenvalue pole placement.

## III. Theory

By picking angles for the gimbal and rotor, and furthermore expressing the model in equilibrium (by setting vdot1 = vdot2 = 0) yields (q2 = n(pi) OR tau3 = 0) and (tau2 = 0). This transforms Equations of Motion(1) to Equations of

Motion(2). Model(2) is used to create A and B matrices. Using any small magnitude for negative eigenvalues (<5 due to actuators in design only providing maximum 5 N(m)) and the matrices found before, K can be found. This K is used in the controller with the new A and B matrices found by utilizing nonlinearized Model(1) since the actual system is Model(1), not Model(2). The controller uses state matrix x where each state variable is replaced with the difference between each state variable and the corresponding equilibrium state value. The input matrix u is set as linearly related to x by factor of matrix K.

f (Model 2) Linearized h (Model 1) Original

$$\begin{bmatrix} v_1 \\ v_2 \\ -\dfrac{1000\tau_3 \sin(q_2)}{10\sin^2(q_2)-511} \\ \dfrac{1000\tau_2}{11} \end{bmatrix} \begin{bmatrix} v_1 \\ v_2 \\ -\dfrac{1000t_3 \sin(q_2)+5v_1v_2\sin(2q_2)+10v_2v_3\cos(q_2)}{10\sin^2(q_2)-511} \\ \dfrac{1000t_2}{11} - \dfrac{10v_1v_3\cos(q_2)}{11} \end{bmatrix}$$

## IV. Procedure

The procedure followed will be broken into subcategories including Equations and Matrices, Correctness Check, and Digging Deeper.

### A. Equations and Matrices

Firstly, vector-valued functions were found by setting up a matrix that utilizes the equations of motion with reduced v1dot and v2dot. This function, f, was lambdified to find the values of v1, v2, v1dot, and v2dot with respect to the equilibrium values. Equilibrium angle for the platform was picked to be pi/2, equilibrium angle for the gimble was picked to be pi/4, and the equilibrium angular velocity of the rotor was given as 10.472 (rad/s). All other values for equilibrium are 0. Next, Amatrix is found as lamdified jacobian of f with respect to v and q. Bmatrix is found as lambdified jacobian of f with respect to tau. To find K, negative eigenvalues are chosen and .gainmatrix element of "scipy.signal.placepoles" with respect to Amatrix, Bmatrix, and eigen is defined. This last step was repeated for the original equations of motion to define nonlinA as lambdified jacobian of h with respect to v and q and nonlinB as lambdified jacobian of h with respect to tau.

### B. Correctness Check

The first check was for the A and B matrices, symbolically, for the linearized relationship. jacobian of f with respect to both q,v and tau were calculated and checked for 4x4 and 4x2 sizes. A was checked for [0,0,1,0],[0,0,0,1] relationships with accurate simplification of v1dot and v2dot. B was checked for [0,0],[0,0] relationships with accurate simplification of v1dot and v2dot. Next, fnum, Anum, and Bnum were calculated as lambdified jacobians to verify the values of Amatrix and Bmatrix. Firstly, fnum was examined to make sure the intuition behind equilibrium and pole placement was correct, which resulted in a 4x1 zeroed matrix. Bmatrix was then calculated as a 4x2 matrix with majority zeroes and two nonzero values. Amatrix was calculated as a 4x4 matrix with majority zeroes and two values of1. These matrices are not the matrices for the final simulation, so the form should be simple compared to nonlinA and nonlinB. For the eigenvalues, computedpoles was used initially to ensure the eigenvalues were being inputed correctly. G was later used as the matrix equivalent of Amatrix-Bmatrix@K, which showed linalg.eigvals as chosen eigen matrix. Anum2 and Bnum2 were created as the lambdified jacobians of h for nonlinA and nonlinB, representing the original vdot relationships, similar to that of Anum and Bnum for Amatrix and Bmatrix. The nonlinA matrix was calculated as very similar to Amatrix, but, as expected, with more nonzero values. The nonlinB matrix was calculated as equivalent to Bmatrix. Furthermore, the linalg.eigvals for the equivalent matrix of nonlinA-nonlinB@K resulted in all negative eigenvalues. This means the use of K for nonlinearized model causes convergence (or asymptotic stability).

### C. Digging Deeper

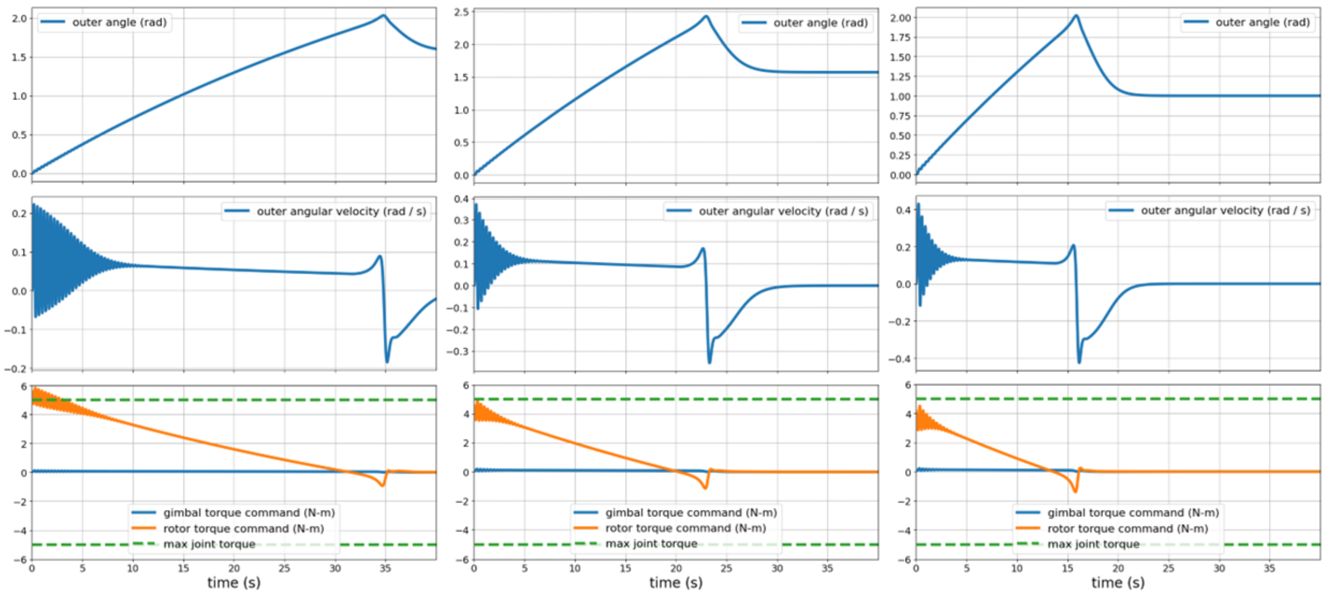
To examine how the change in angles chosen for equilibrium change the results of the controller for the nonlinearized system, q1 and q2 equlibrium points were changed from the original q1 = pi/2 > q2 = pi/4 to q1 = pi/2 = q2 and q1 = 1 < q2 = 2. The differences in peak time and overshoot were noted. To examine how the change in eigenvalues chosen change the results of the controller for the nonlinearized system, eig = [-4.5, -3, -1.5, -1] was changed to eig = [-5,-4,-3,-2] and eig = [-1,-2,-3,-4] noting that the size of the eigenvalues must be 4x1 and all must be negative (to assure asymptotic stability) and have magnitude<5 (to assure conditions of system are met). Differences in peak times and initial rotor torque were noted.

## V. Results

The controller was set up with an x matrix of size 4x1, each value representing the difference of between the variable and related equilibrium value where u = -K@x. The torque applied to the gimbal is 0 on both components while torque applied to rotor has first component. This results in 3 plots, using RobotController and Plot code given. The first plot represents the angle of the platform with respect to time, converging to chosen q1 equilibrium. The second plot represents the angular velcoity with respect to time, converging to 0. The third plot represents the torque on the gimbal and rotor both converging to 0. Following the described setup with extended run time (20s), the simulation runs with no error and shows platform reaching angle = pi/2 and gimbal reaching angle = pi/4 with inital angular velocity = 100 rev/min.

### A. Change in Equilibrium Angles

The initial equilibrium angles used resulted in the first set of graphs shown with angle converging to q1-1(equilibrium) = pi/2, angular velocity converging to 0, and torques for both gimbal and rotor converging to 0. The gimbal torque was consistently 0 while the rotor torque started large and converged to the proper value of 0. Angle, angular velocity, and rotor torque all overshot just before 35 seconds and reached relative convergence by 40 seconds. It is important to note that the rotor torque begins oscillating just outside of the maximum torque range. The altered equilibrium angles of q1-2 = q1-1 = pi/2 = q2-2 yield a significantly more stable and faster convergence for angle, angular velocity, and rotor torque. All three values overshoot at about 23 seconds and reached relative convergence by 30 seconds. It is important to note that the rotor torque never oscillates outside the maximum range of torque. The altered equilibrium angles of q1-3 = 1 and q2-3 = 2 yield an even more stable and faster convergence for angle, angular velocity, and rotor torque. All three values overshoot by about 15 seconds and reached relative convergence by 20 seconds. It is important to note the rotor torque approximately stays under 4, which is less stress on the torque command.



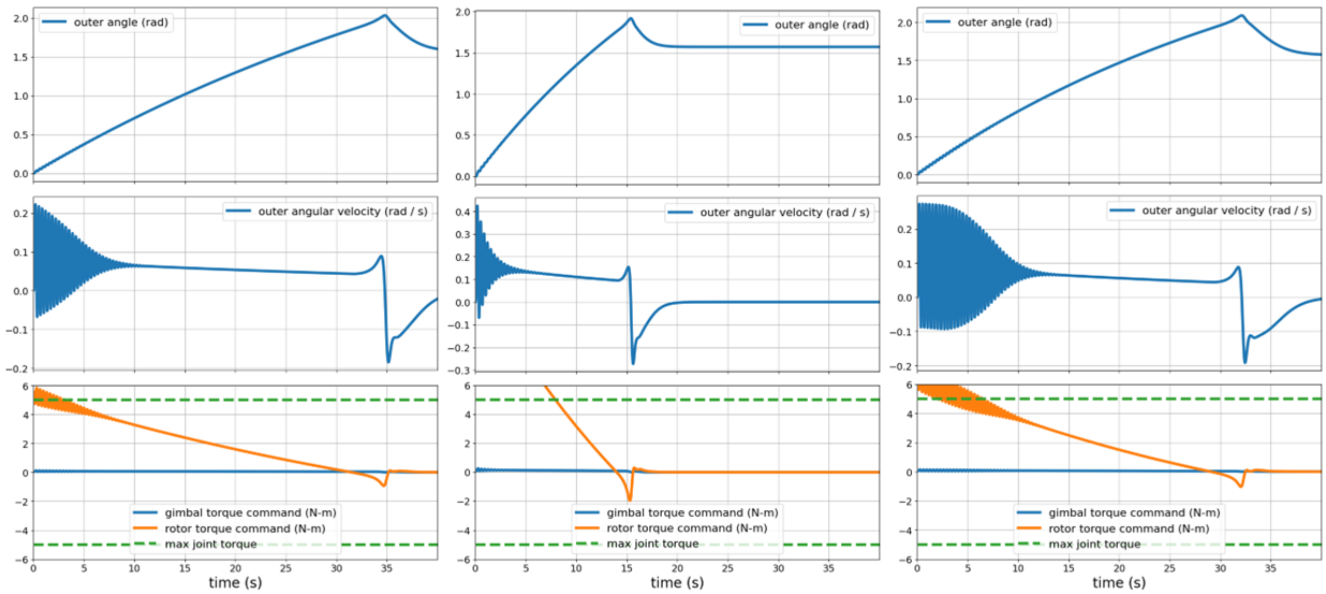

q_1e = np.pi/2, q_2e = np.pi/4, v_1e = 0, v_2e = 0, v_3e = 10.472, q_3e = 0, tau_1e = 0, tau_2e = 0, tau_3e = 0

q_1e = np.pi/2, q_2e = np.pi/2, v_1e = 0, v_2e = 0, v_3e = 10.472, q_3e = 0, tau_1e = 0, tau_2e = 0, tau_3e = 0

q_1e = 1, q_2e = 2, v_1e = 0, v_2e = 0, v_3e = 10.472, q_3e = 0, tau_1e = 0, tau_2e = 0, tau_3e = 0

**B. Change in Eigenvalues**

The initial eigenvalues [-4.5, -3, -1.5, -1] yield an overshoot peak at about 35 seconds and reach relative convergence by 40 seconds. It is important to notice that the rotor torque starts oscillating outside of the max torque range and angular velocity oscillates very fast in the beginning with a relatively steady angle climb. The altered eigenvalues [-5, -4, -3, -2] yield an overshoot peak at about 15 seconds and reach relative convergence by 17.5 seconds. It is important to notice that the rotor torque starts oscillating well beyond the limit, angular velocity converges faster, and angle q1 increases much more steeply. The altered eigenvalues [-1, -2, -3, -4] yield an overshoot peak at about 32 seconds and reach relative convergence by 37.5 seconds. It is important to note that the rotor torque starts just outside the limit and has more oscillation outside of the bounds than the first.




Eigenvalues: [-4.5, -3, -1.5, -1]

Eigenvalues: [-5, -4, -3, -2]

Eigenvalues: [-1, -2, -3, -4]

# VI. Conclusion

When viewing changes in initial equilibrium angle, there are many changes in how the controller processes changes in the nonlinearized system. For example, as the angle of the gimbal increases and surpasses the angle of the platform, convergence for the system involving platform angle, angular velocity, and rotor torque happens faster with more precision. The rotor torque stays within range and reaches 0 faster, the angular velocity converges sooner with consistent overshoot, and platform angle reaches q1(equilibrium) faster. There are many changes in the effect of the controller due to eigenvalue placement as well. For example, as the eigenvalues increase in magnitude, angle of the platform, angular velocity, and rotor torque converge faster. As eigenvalues increase, rotor torque initially oscillates more out of the maximum range. For all examples, the controller made the system converge to the desirable state defined.