AA279c Project Report: Modeling the ISS

Toby J. Buckley

June 8, 2017

testing

1 Problem Set 6 - Attitude Determination with Sensor Errors

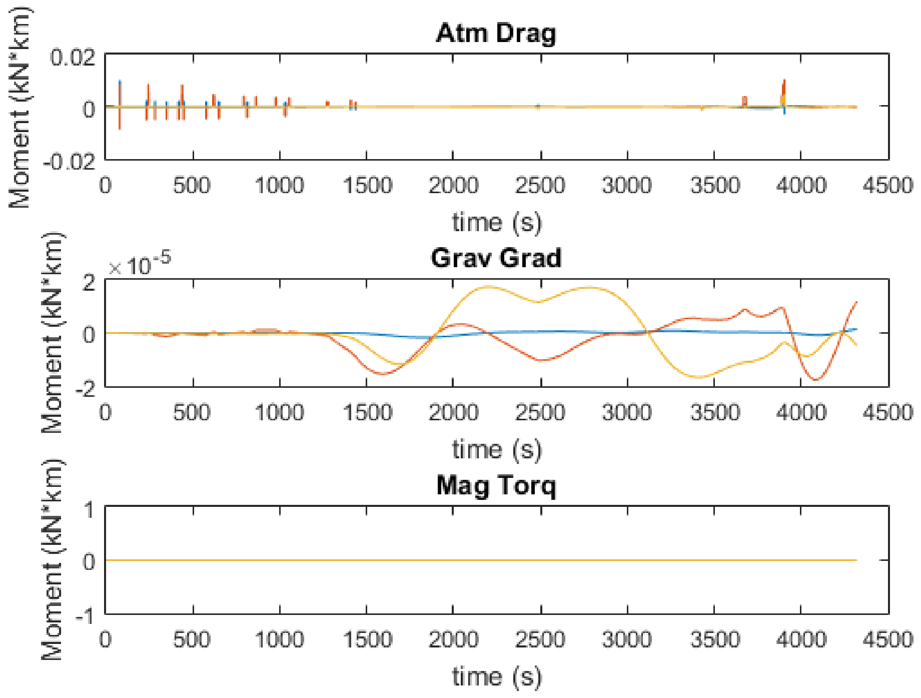

Figure 1: EKF performance.

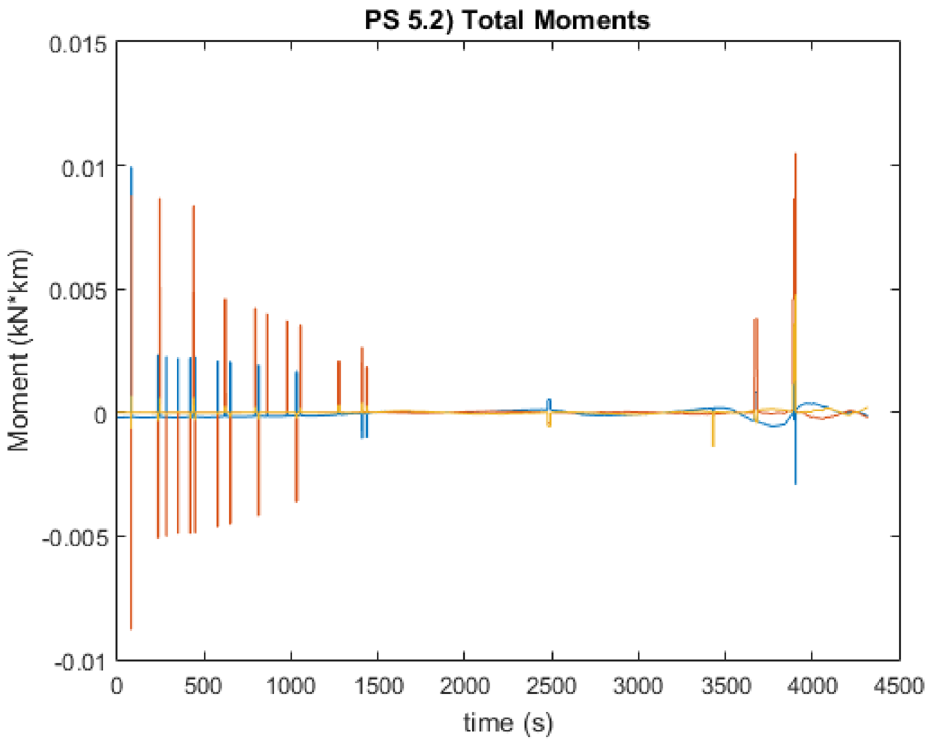

Figure 2: EKF performance.

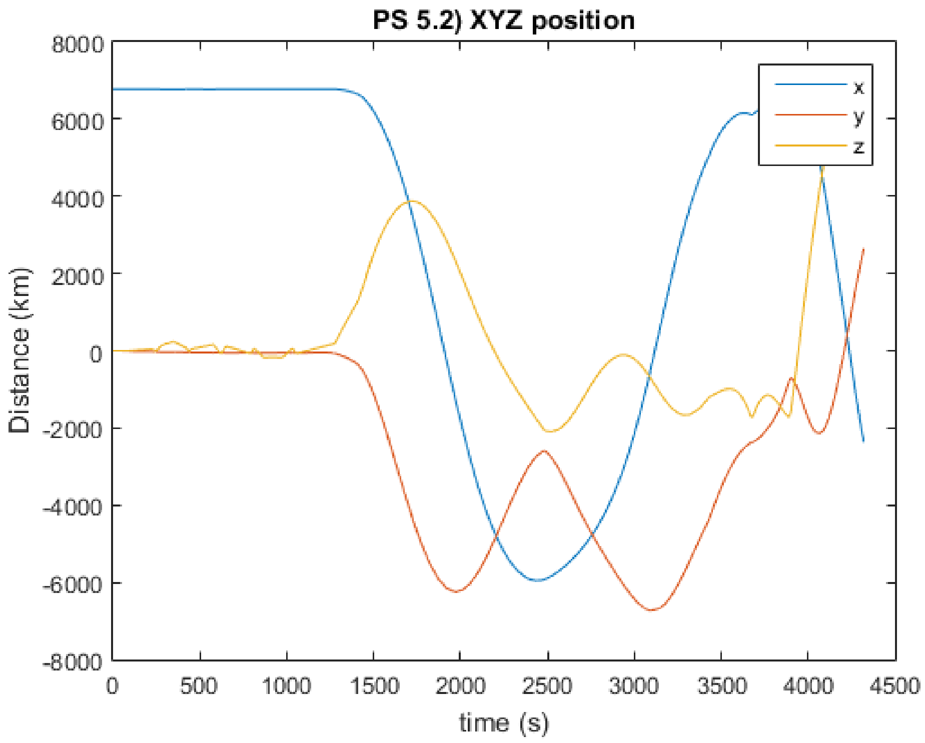

Figure 3: EKF performance.

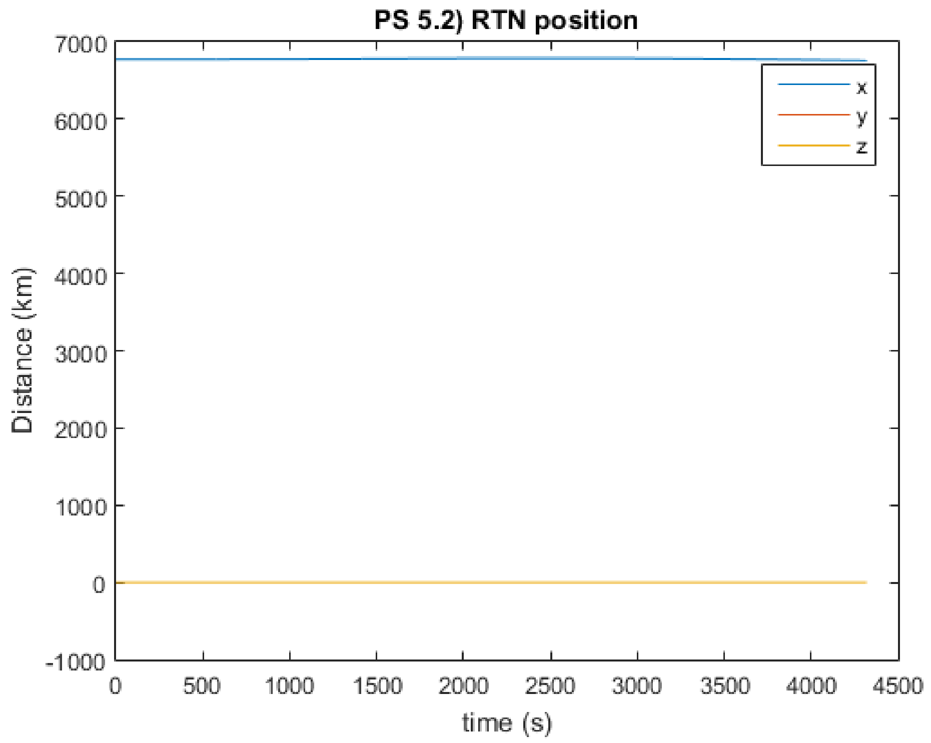

Figure 4: EKF performance.

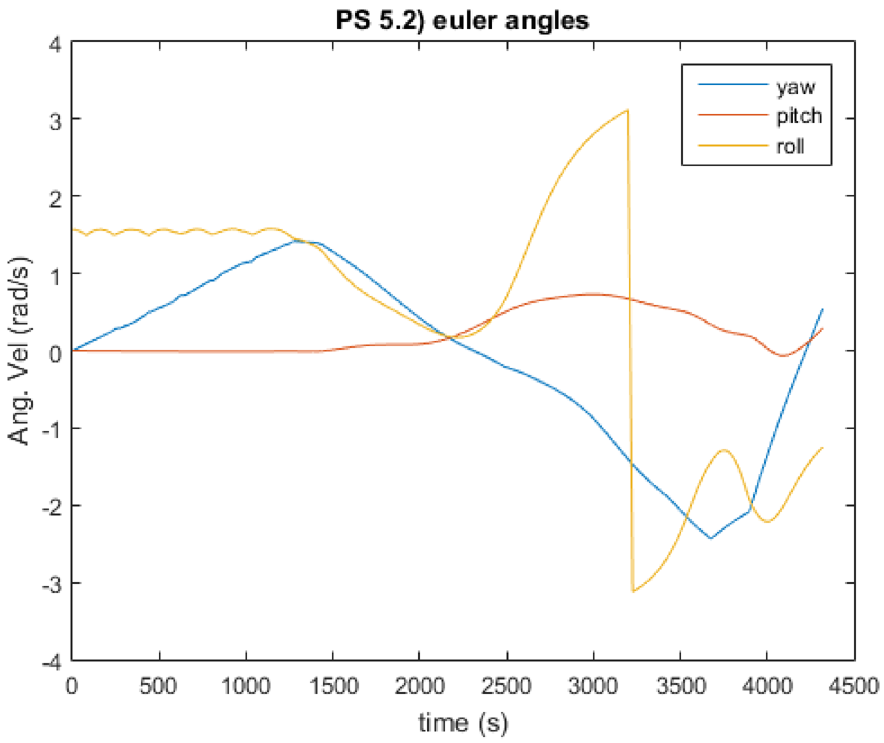

Figure 5: EKF performance.

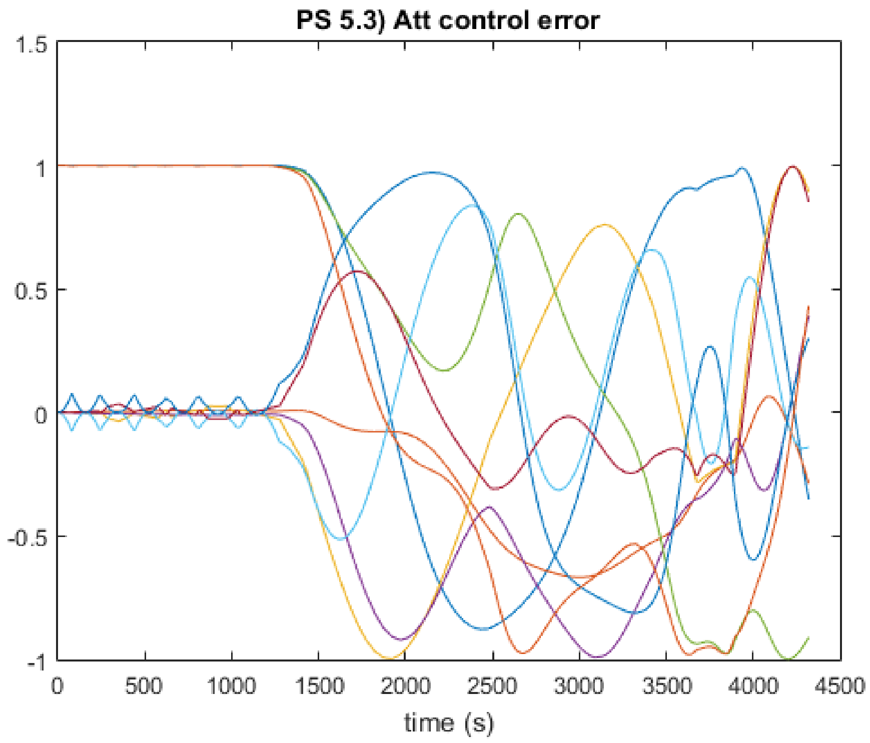

Figure 6: EKF performance.

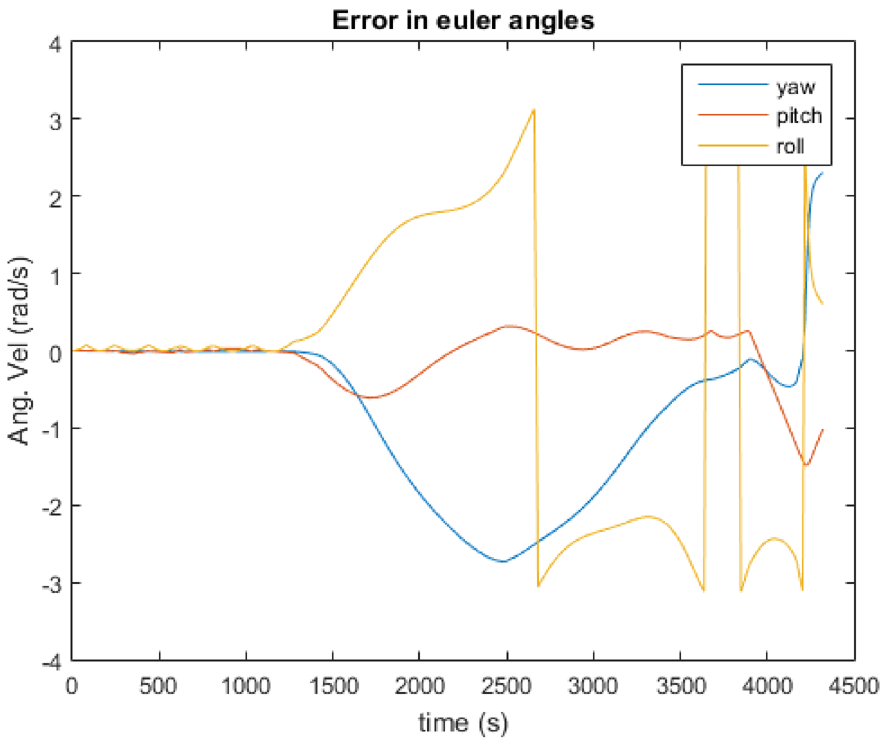

Figure 7: EKF performance.

2 Problem Set 6 - Attitude Determination with Sensor Errors

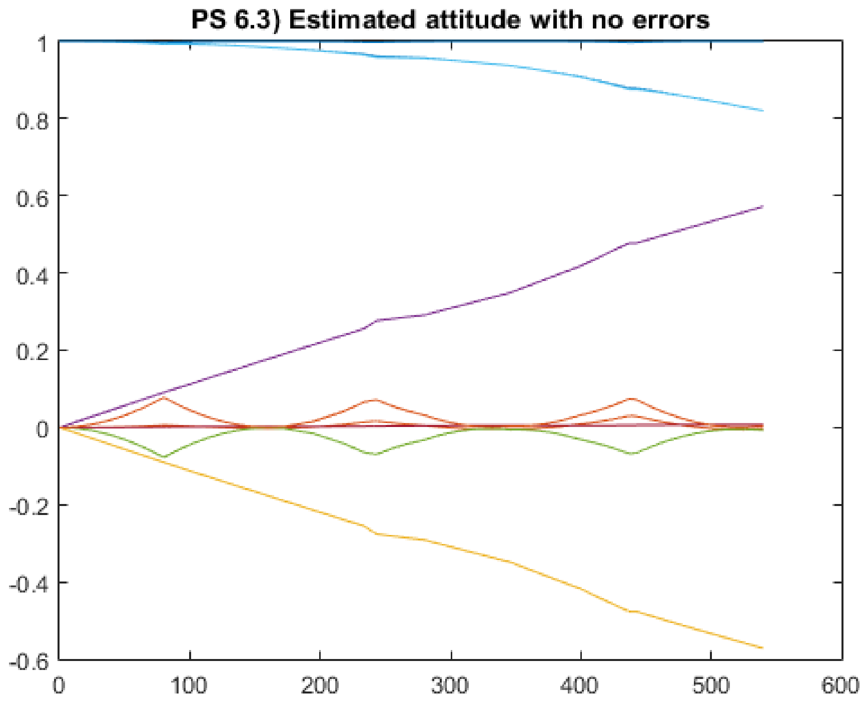

Figure 8: EKF performance.

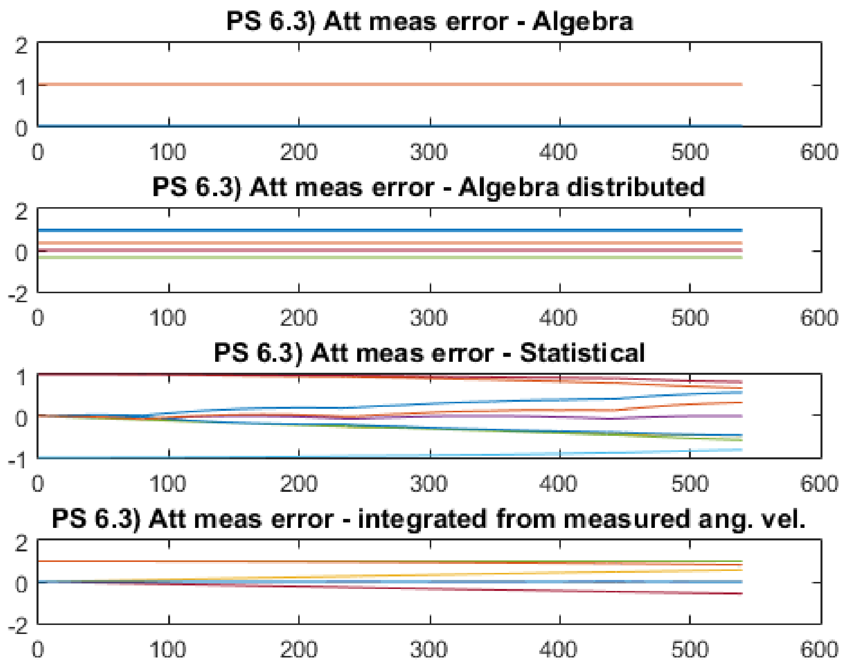

Figure 9: EKF performance.

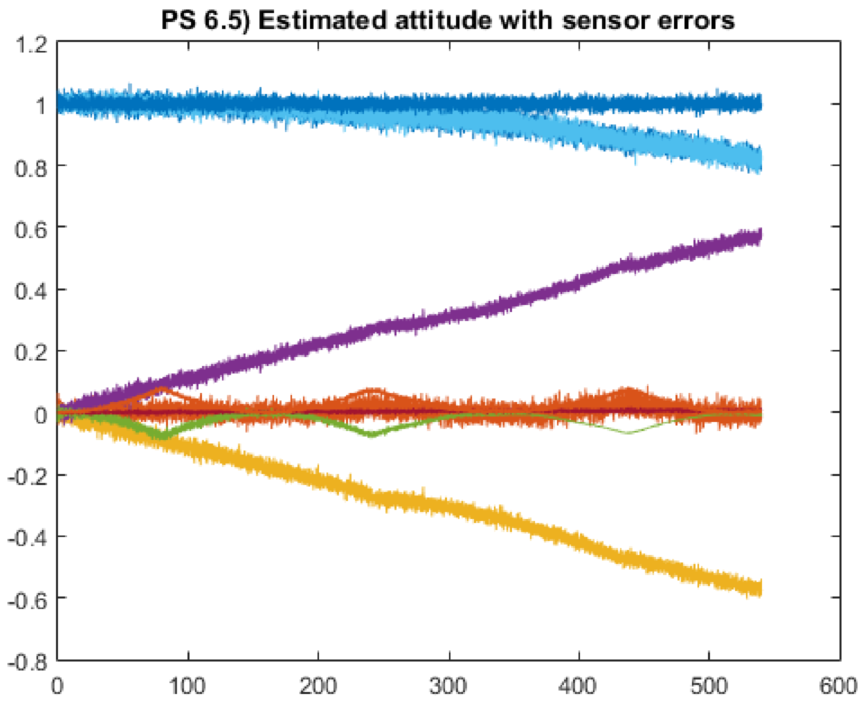

Figure 10: EKF performance.

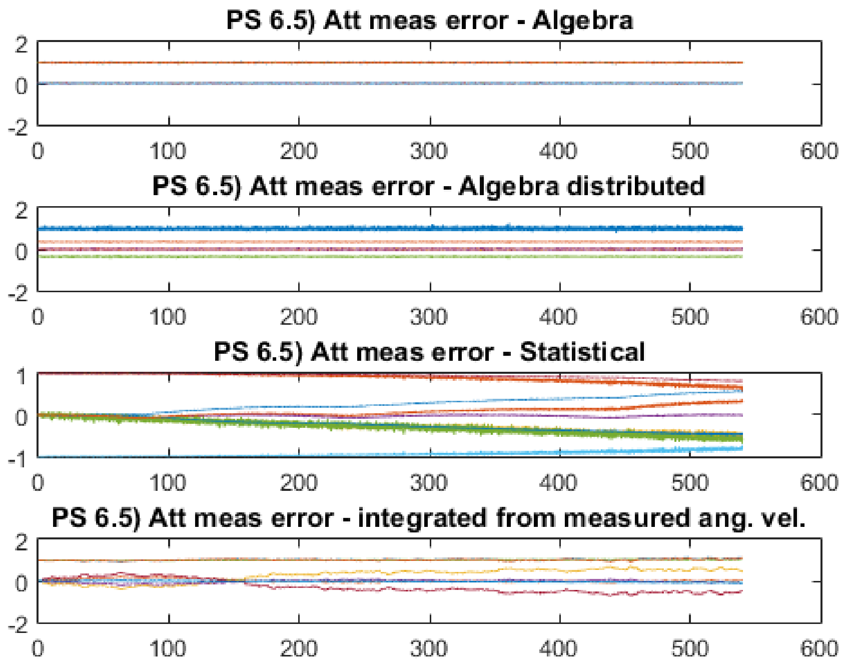

Figure 11: EKF performance.

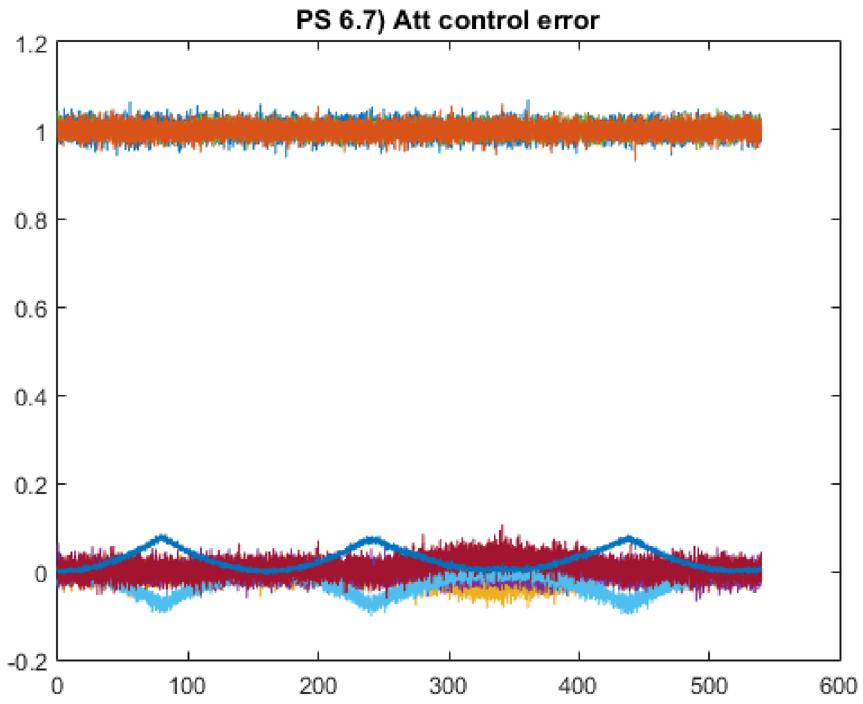

Figure 12: EKF performance.

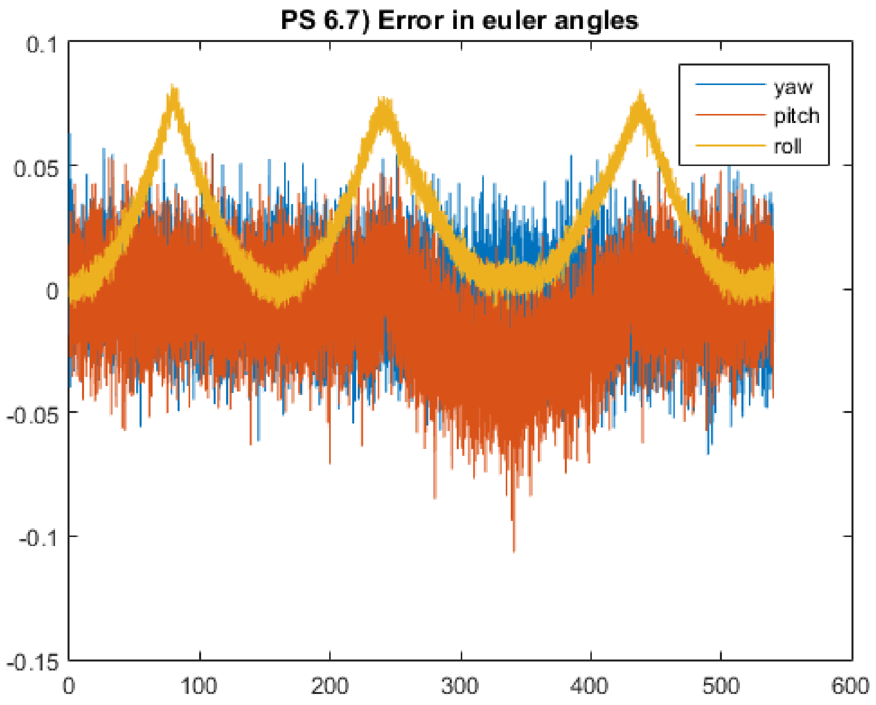

Figure 13: EKF performance.

3 Problem Set 7 - Implement Extended Kalman Filter

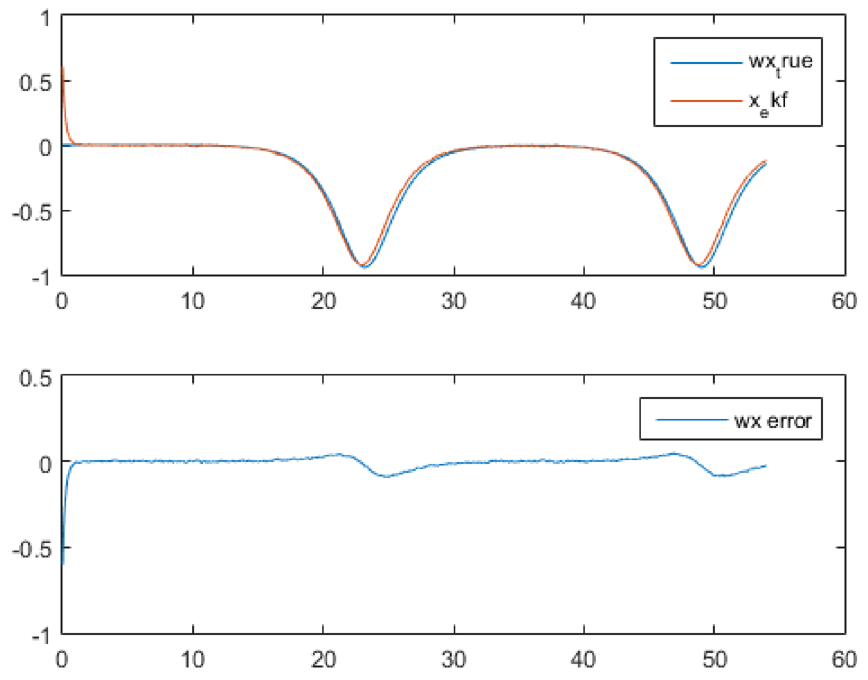

Figure 14: EKF performance.

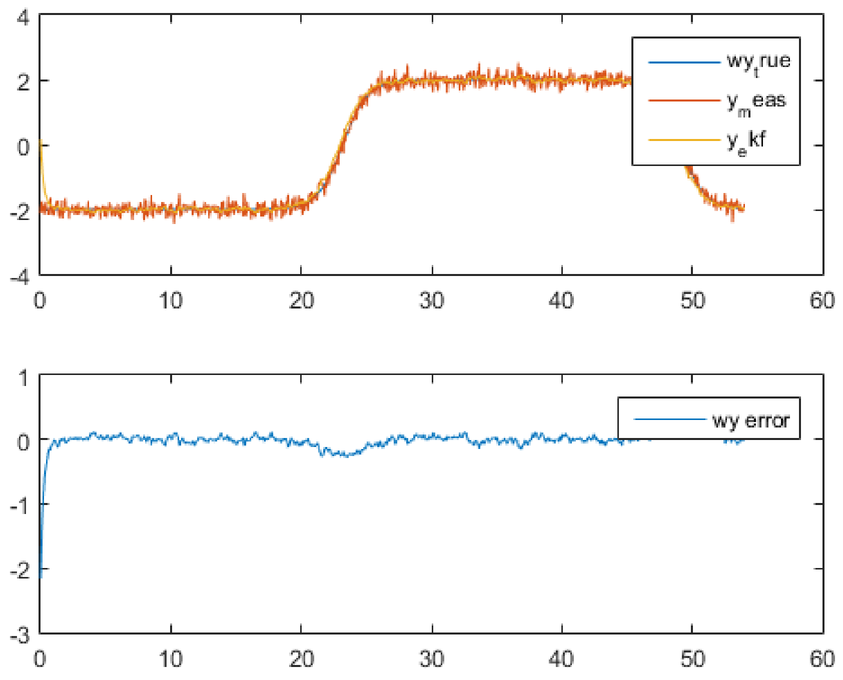

Figure 15: EKF performance.

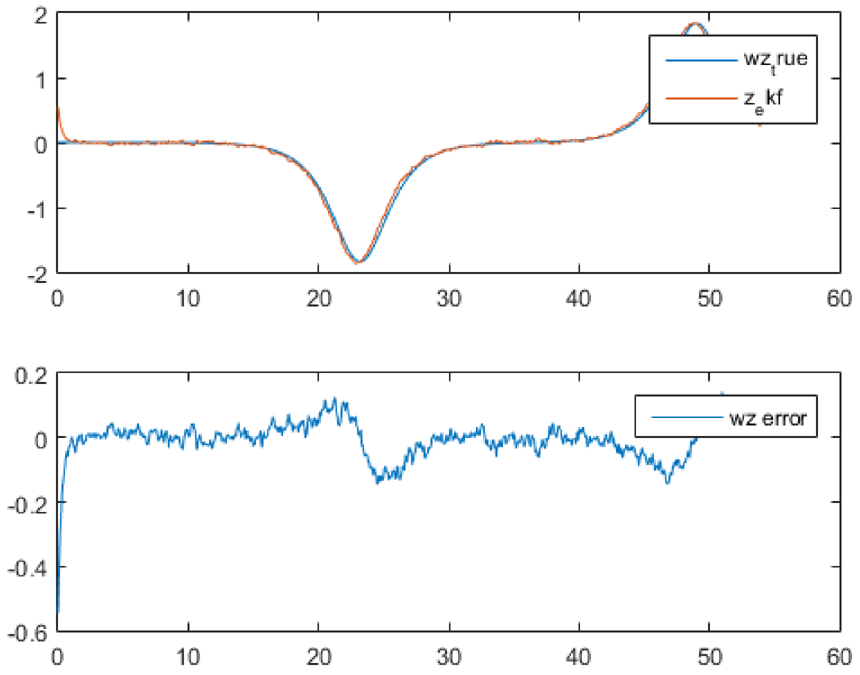

Figure 16: EKF performance.

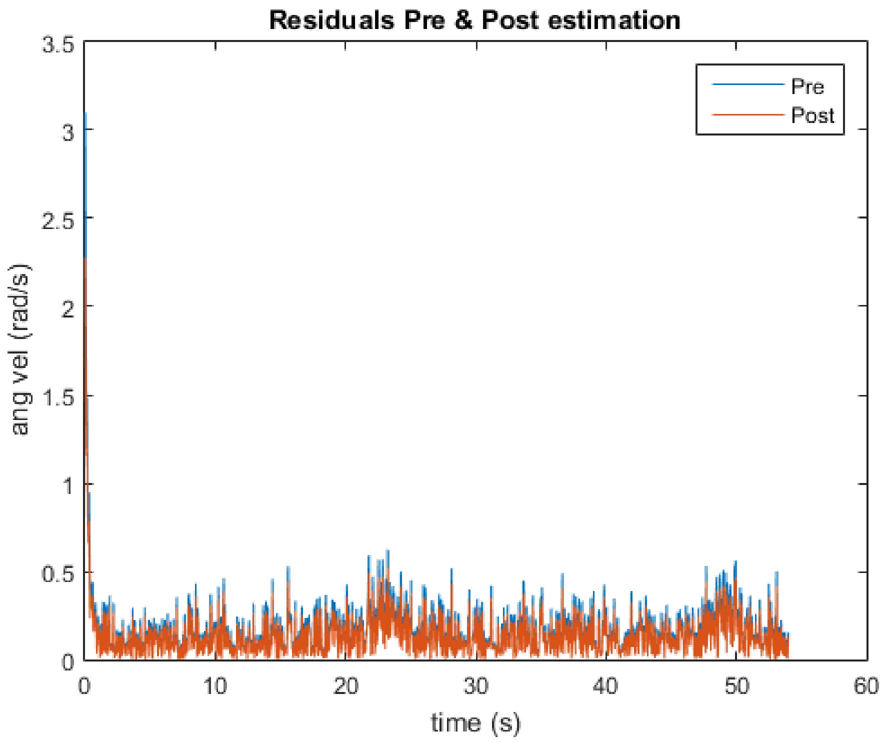

Figure 17: EKF performance.

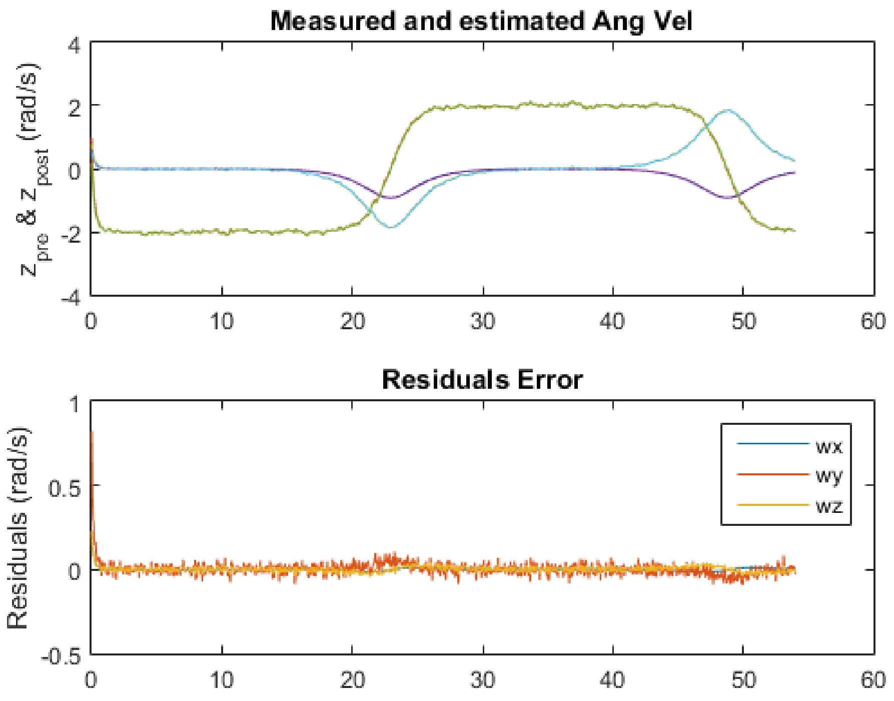

Figure 18: EKF performance.

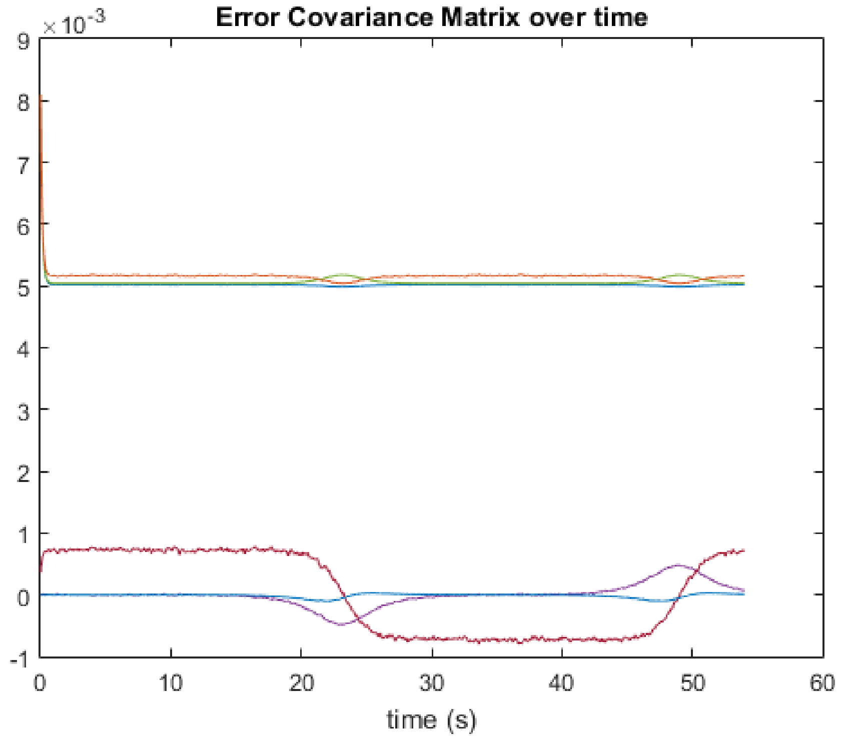

Figure 19: EKF performance.

4 Problem Set 8 - Implement Actuators and Controllers

The International Space Station uses 4 control-moment-gyros it uses for day-to-day station keeping. Each CMG can produce 4760 N*m*s, or 258 N*m, and spins at 691 rad/s. In reality, the ISS has all four aligned the in the same direction, but for simplicity I've decided to model three CMG's each aligned in a different body axis direction.

The control moment gyros operate by running at a fixed very high speed rotation, with large mass and radius. By gimbaling the CMG, the angular momentum vector is shifted which causes a reaction torque on the rest of the satellite due to the conservation of angular momentum.

$$I\dot{\omega} + \omega x I\omega = M + M_c;\ M_c = -A\dot{L_w} - \dot{A}L_w - \omega x A L_w \qquad (1)$$

Because we have changed the dynamics of the vehicle by adding additional large rotating masses, the Euler equations must also change. Eq. 1 shows how the CMG's are taken into account, by grouping them together into a control moment term, which we are able to take advantage of when determining a control sequence.

Control of the aircraft is performed using two successive loops (Fig. 13). First, a high-level objective attitude is created (example: follow the RTN frame) and passed to the attitude controller. A first controller generates commanded angular velocity, which is passed to the second controller. The output of the inner-loop are the desired moments that the CMG's must impart on the vehicle. Those are perturbed slightly by assuming imperfect actuators, and then passed into the environment simulator. The required CMG angles are computed from the control-loop output by solving for $\dot{A}$ and performing a 1st order integration.

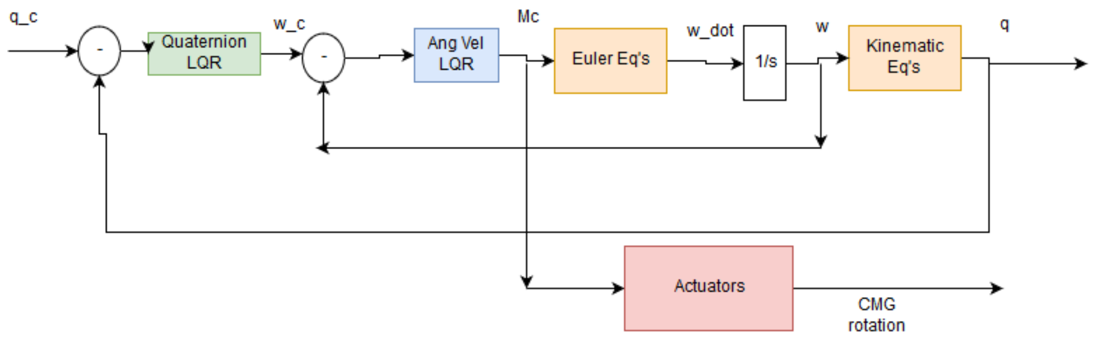

Figure 20: Block diagram for the controller used onboard the satellite. Two LQR's are utilized, one for attitude and the other for a lower-level ang. vel. controller.

The block diagram for the controller (Fig. 13) shows how the different controllers pass information to each-other and the environment. In order to obtain the controller gains, a linear-quadratic controller is implemented.

$$x_{k+1} = Ax_k + Bu_k \tag{2}$$

$$u = -Kx \tag{3}$$

The linear-quadratic controller, or LQR, is well-documented and used widely in the controls community. Eq. 2 shows the discrete state space model, where the next state is a linear function of the current state and controls. It's assumed a linear gain is applied to the state and used to control the system (Eq. 3). By setting a weight matrix for the state (Q), for the control (R), and for the combination state and control (N), an optimal gain matrix can be solved for which minimizes the cost of regulating the system (shown below). The matlab function $K = dlqr(A, B, Q, R)$ is used to obtain the gain matrix.

$$J = x_N^T Q x_N + \sum_{k=0}^{N-1} \left(x_k^T Q x_k + u_k^T R u_k + 2 x_k^T N u_k \right)$$

The plant matrix, A, was obtained by linearizing the euler and kinematic equations. Taylor series expansion (shown below) was used to obtain a first order approximation of the differential equations.

$$f(x) \approx f(a) + f'(a)(x-a)$$

For attitude, $\dot{q} = 1/2 \Omega q$ was linearized with respect to angular velocity. Below is the jacobian of q, as well as the discrete next-step quaternion.

$$\nabla_\omega q = \begin{bmatrix} q4 & -q3 & q2 \\ q3 & q4 & -q1 \\ -q2 & q1 & q4 \\ -q1 & -q2 & -q3 \end{bmatrix} ; \; q_{k+1} \approx dt * 1/2 * \nabla_\omega q|_q * \omega + q_k$$

Eq. 4 places the known quantities into the usual discrete linear system form. For the inner-loop, the same linear model used in the EKF is used here.

$$q_{k+1} = A q_k + B\omega; \; A = eye(4), \; B = dt * 1/2 * \nabla_\omega q|_q \tag{4}$$

In the simulink model, the control loop is ran once every ten time-steps. This is valid because linearized dynamics are accurate within an area surrounding the point of linearization. Satellite's dynamics evolve relatively slow compared to the time-step, so the gain matrix is only updated sparingly.

Initially, I had significant trouble getting the LQR controllers to function properly. The satellite could control its ang. vel., but not the attitude. After careful inspection, I figured out the issue: the weight matrices on each LQR were not properly adjusted after implementing the successive control-loop structure. With no feed-forward terms or direct coupling between attitude and control moment, there is a natural delay whenever an attitude is commanded. If the attitude controller doesn't allow enough time for the ang. vel. to 'catch up', then it will change the commanded ang. vel. before anything even happens. This was resolved by weighting the attitude rather low, and the ang. vel. higher, so that the inner-loop reaches the commanded velocity much quicker and is able to affect the attitude as desired.

In testing the controller, a nominal trajectory was defined for the ISS to follow. The satellite begins in the RTN frame with $\omega = [0 - n00]'$ and is subject to perturbations. The goal is to maintain the desired attitude.

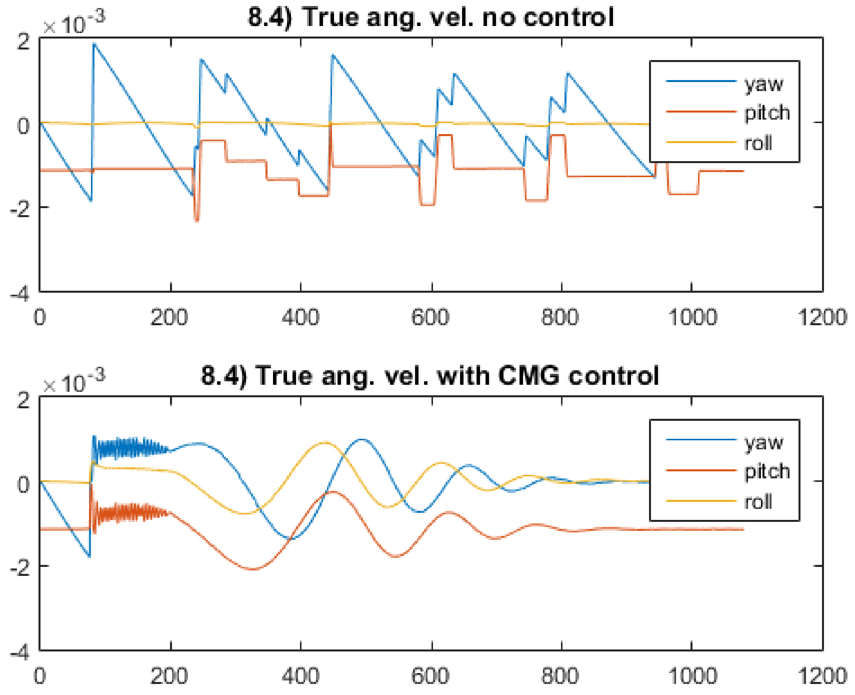

Figure 21: Angular velocity with and without control.

Fig. 18 shows how the euler angles evolve during the flight. With no control the yaw and pitch vary sinusoidally due to the perturbations present.

With active control the euler angles converge onto their nominal value and retain them for the rest of the flight. The jitter in the beginning is due to the EKF's random initialization. It hasn't yet converged onto the true value so when $\omega_{measured}$ is fed into the controller it produces garbage control moments.

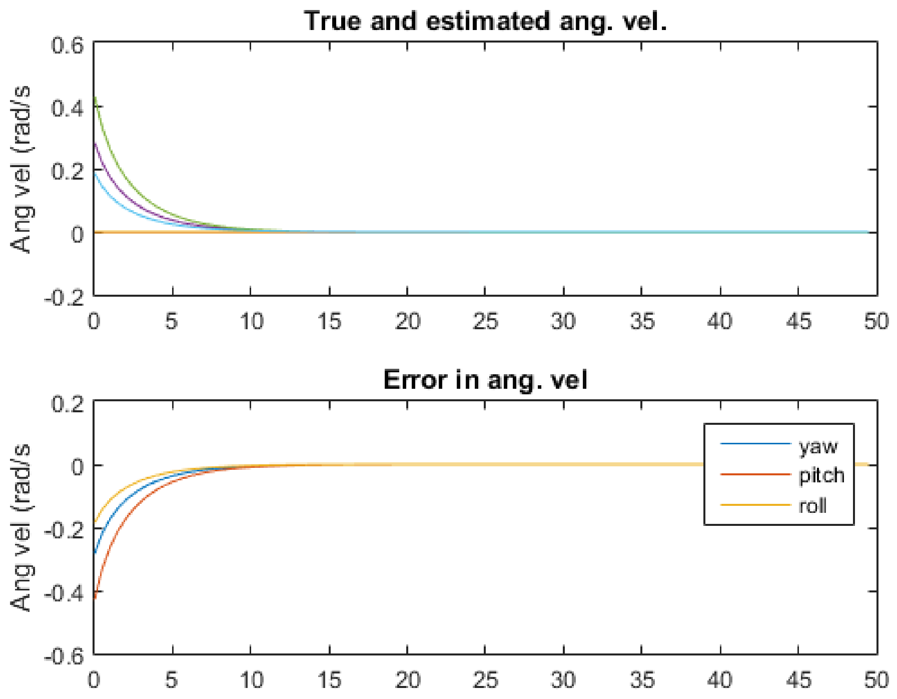

Figure 22: EKF performance.

Fig. 15 shows how the extended kalman filter behaves near the beginning of the flight. After randomly initializing the state, new observations allow the EKF to quickly determine the optimal P matrix and converge onto the true observations. Within 10 seconds, the error in angular velocity approaches zero.

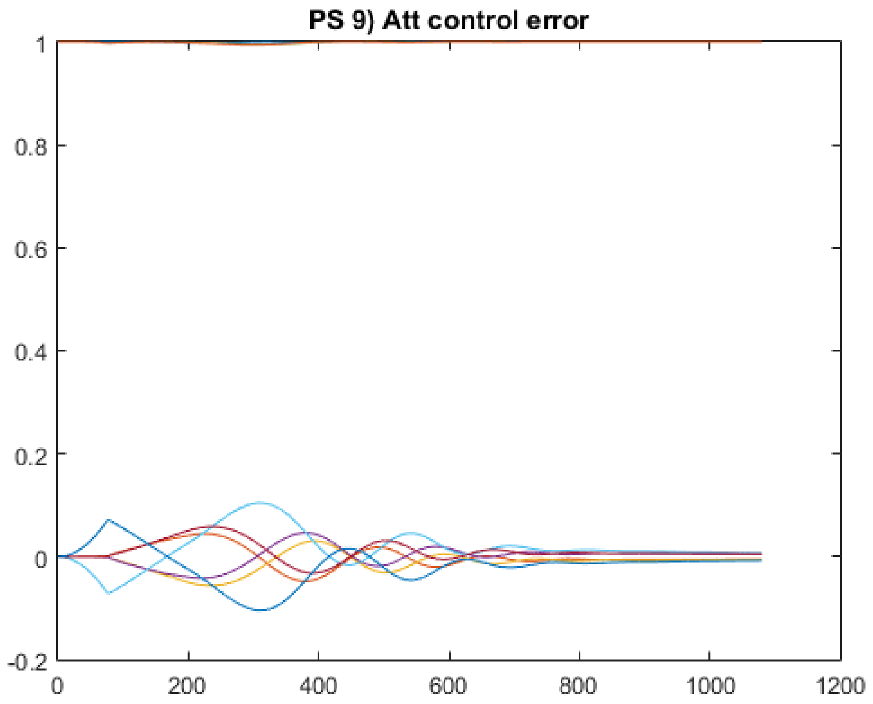

Figure 23: Error in attitude control between principle and RTN frames.

Above is the attitude control error during the flight. This represents the direction cosine matrix between principle axes and RTN frame. Perturbations and state uncertainty lead to some drift between the frames, but the active controller quickly overcomes them and returns the ISS to its nominal attitude.

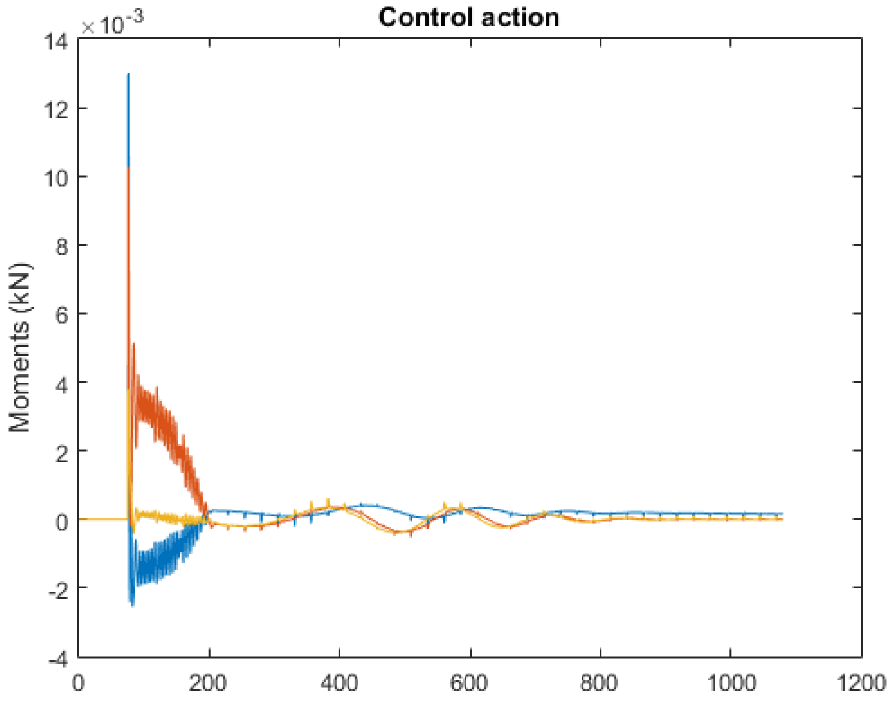

Figure 24: Angular velocity while de-tumbling.

The output of the controller is how much moment to impart in each principle direction. Besides the initial jitter, very little moment is required to keep the orbit.

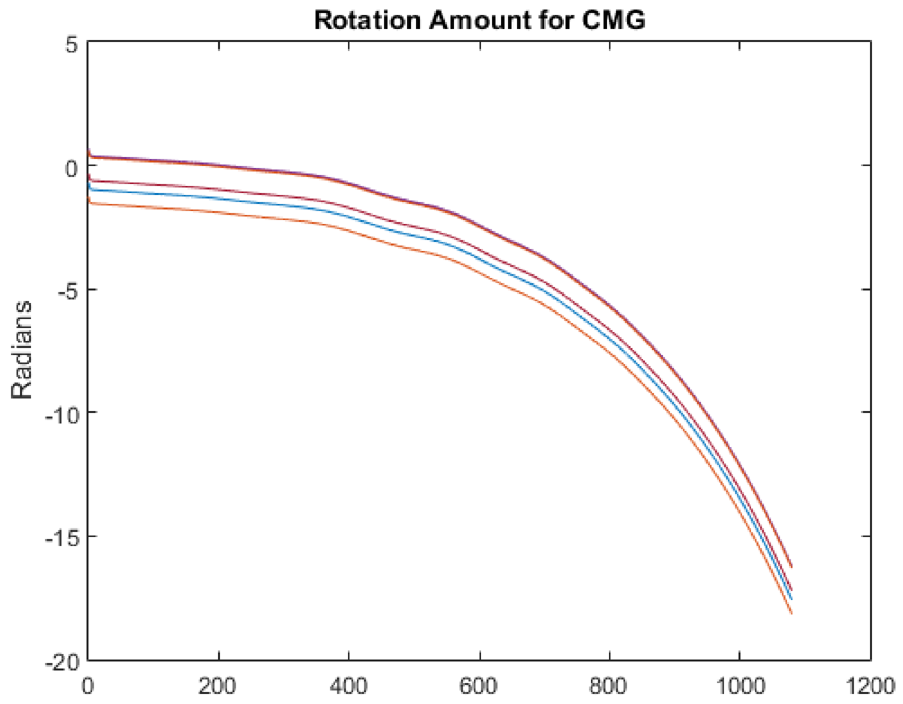

Figure 25: Angular velocity while de-tumbling.

After control moments are generated, the amount of gimbal required to provide them is calculated. Above shows how the combined CMG system must rotate in order to satisfy the controller's requirements. This result is troubling, as each CMG needs to rotate multiple revolutions to keep the station for part of an orbit. I suspect a bug or a sizing issue.

5 Problem Set 9 - Define and Execute a Slew Maneuver

A problem all satellites deal with is de-tumbling. When launched into orbit the satellite is harshly ejected from its rocket into a target orbit and must immediately begin to align itself into its nominal position. I've simulated a tumbling environment by setting $w0 = rand(3,1)$ and allowing the dynamics and controller to stabilize the system naturally. The target attitude for the de-tumble is the earth-facing RTN frame.

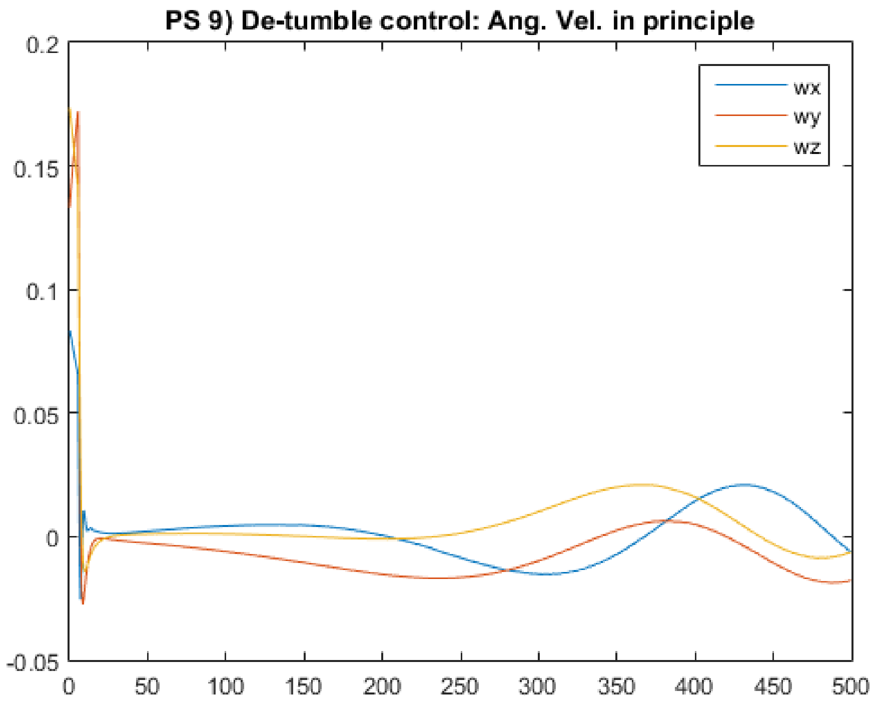

Figure 26: Angular velocity while de-tumbling.

Fig. 19 shows the angular velocity while de-tumbling. Because of the high priority on matching w to $w_{desired}$ the velocities stabilize quickly. After that all changes are due to the attitude controller commanding certain angular velocity in order to align itself with the RTN frame.

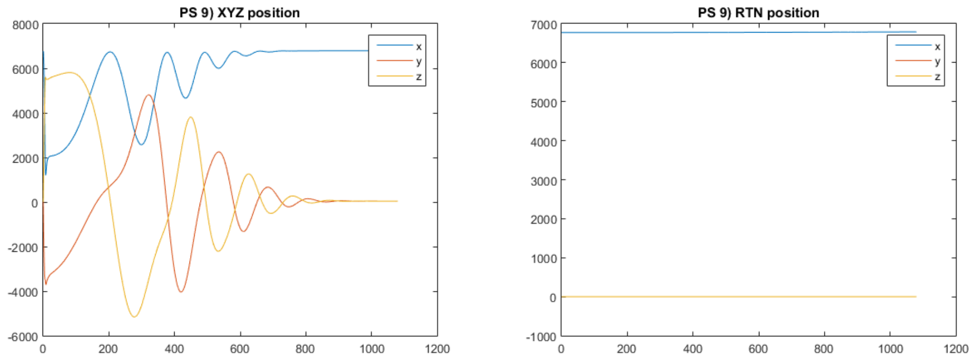

Figure 27: The position during a de-tumble. The Target is earth-facing in the RTN frame.

Fig. 20 shows how the principle axes aligns itself with the RTN frame over time. Oscillations are present due to the sinusoidal dynamics of the rotation cosine matrices.

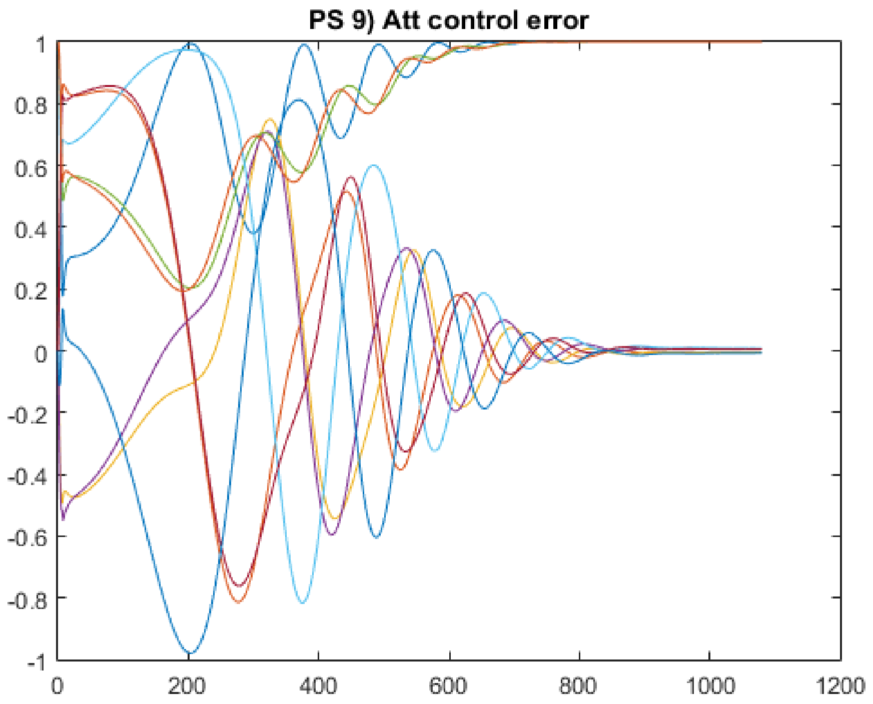

Figure 28: Attitude control error until convergence.

Fig 21 shows the evolution of the attitude control error over time. It's clear the the tumbling has a huge effect on the relative alignment of the principle axes with the desired RTN frame. But this problem is overcome within 800 seconds of ejection with the satellite's onboard controller.