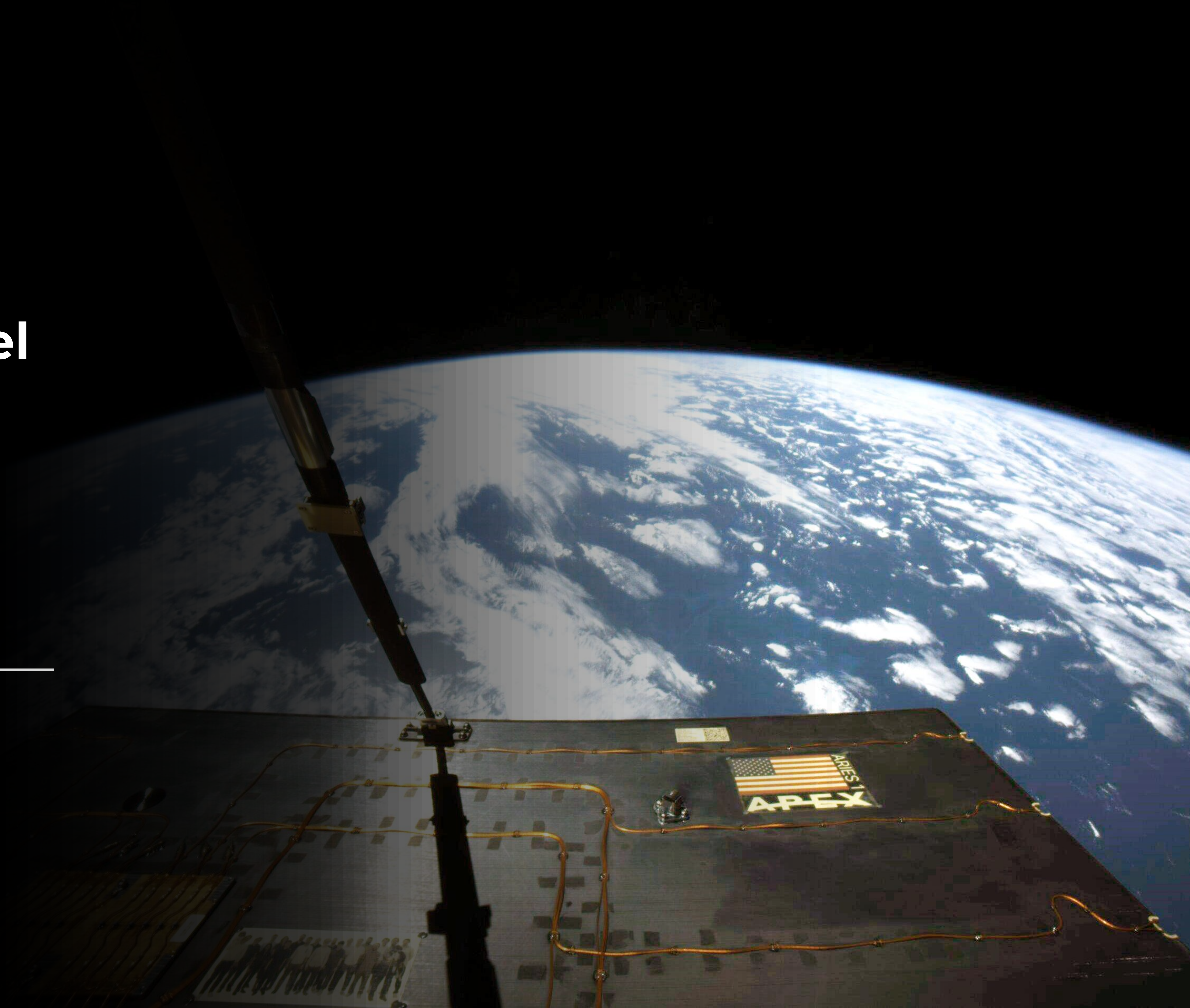
Aries Precision Controller Retuning with Reaction Wheel Time Delay

Tycho Bogdanowitsch

8/8/25

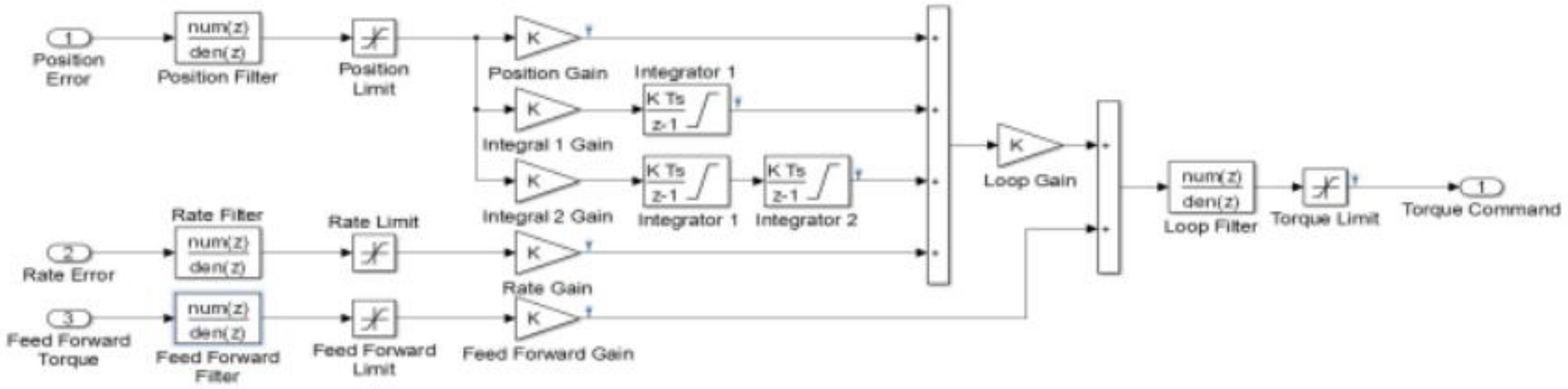


Background

- Reaction wheel actuation delay of up to ~250 ms
- Standard and precision controller run at 10 Hz (0.1 sec / cycle)
  - Conservative estimate of maximum delay of 3 cycles (300 ms)
- Last time – precision controller with current gains
  - No delay case
    - GM = 12.2 dB (ZOH), 12.5 dB (Tustin)
    - PM = 43.8 deg (ZOH), 46.6 dB (Tustin)
  - 3 cycles of delay case
    - GM = 8.31 dB (ZOH), 9.37 dB (Tustin)
    - PM = 27 deg (ZOH), 29.8 deg (Tustin)
- Goal: With delay included, **margins of 8 dB and 45 deg & 5% settling time of 10 seconds (ISI)**

- ZOH → more representative of MAX implementation on spacecraft (plant)
- Tustin → preserves phase better, used for frequency domain performance (controller)
- MATLAB time delays
  - Continuous → 5^{th} order Pade approximation
  - Discrete → (Z)^(-k), k = # of samples

MAX Controller: PIID

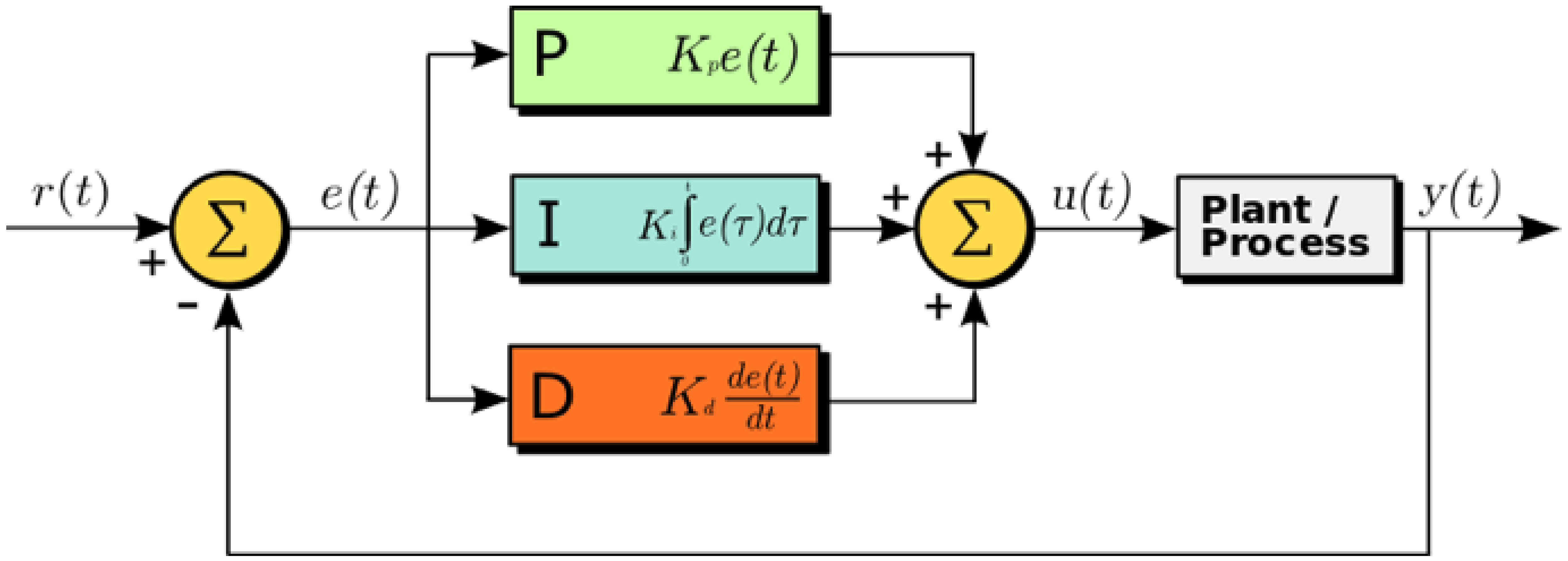



```
// Compute Delta-Quaternion from Previous Cycle to Current Cycle.
QUATERNION  qDelta = Qinv_Q_Mult(qcmd_CBI2BDY[0], qcmd_CBI2BDY[1]);

// Calculate Delta Angles using small-angle approximation.
VECTOR deltaAng_rd = 2.0 * Sign(qDelta[3]) * VECTOR(qDelta[0], qDelta[1], qDelta[2]);

// Determine Rate and Accel from delta Angle and delta T.
m_Data.WCmd_BDY_rdps = deltaAng_rd / thisDeltaTime_s;
```

Rate command (and error) is calculated through finite differencing of position command

Rate command (and error) is calculated through finite differencing of position command

Previous Implementation: Basic PID



$$u(t) = K_p\,\theta_e(t) + K_i \int_0^t \theta_e(\tau)\,d\tau + K_d\,\frac{d\theta_e(t)}{dt}$$

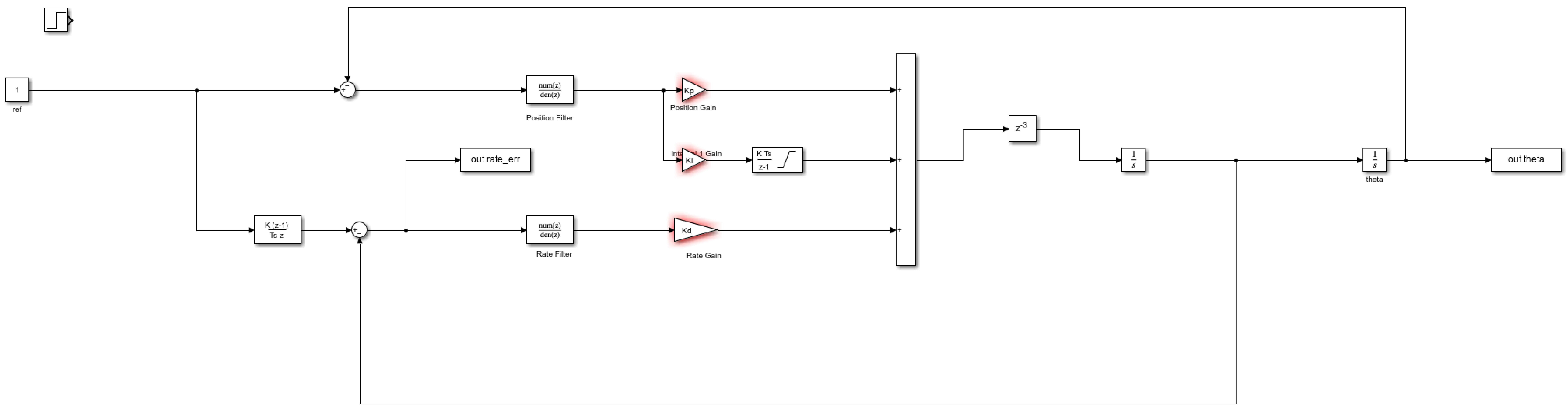
$$\theta_e(t) = \theta_{\text{cmd}}(t) - \theta_{\text{meas}}(t)$$

```
% 10 Hz controller
Ts = 0.1; % sec

% Discrete Low-pass filter
lpfilter = tf([ 8.507829e-2 1.7015657e-1 8.50782e-2 ],[ 1.0e+0 -9.1530955e-1 2.5562269e-1 ], Ts);

% Continuous precision PID controller
C = tf(0.5, 1) + tf(0.1, [1 0]) + tf([0.95 0], 1);

% Continuous double-integrator plant model
P = tf(1, [1 0 0]);

% Open-loop discrete system wihout delay
discPC = c2d(C * P, Ts, 'zoh') * lpfilter;
```

Rate error is not represented well

Instead K_d acts on derivative of position error

New Implementation: PID with Rate Feedback



$$u(t) = K_p\,\theta_e(t) + K_i \int_0^t \theta_e(\tau)\,d\tau + K_d\,\omega_e(t)$$

$$\omega_e(t) = \omega_{\text{cmd}}(t) - \omega_{\text{meas}}(t)$$

$$\omega_{\text{cmd}}(t) = \frac{\Delta\theta_{\text{cmd}}(t)}{\Delta t}$$

Rate error is more representative of MAX implementation

Calculated from difference in rate command and measured rate from plant

Rate command calculated from finite differencing (discrete differentiation) as in MAX

Retuning Script

```
Kp_list = linspace(0.25, 1.25, 21);
Ki_list = linspace(0.005, 0.3, 21);
Kd_list = linspace(0.75, 1.5, 21);
```

- Current gains on spacecraft now: Kp = 0.5, Ki = 0.1, Kd = 0.95
- Leave low-pass filters the same
- Iterate through combinations of Kp, Ki, Kd centered on current gains
- Linearize Simulink model to get open-loop transfer function
- Calculate GM, PM, wgc, wpc using margin()
- Calculate 5% settling time (Ts5) for step response of closed-loop transfer function
- Assign cost to each candidate by comparing to desired GM, PM and Ts5
  - Could add weights
- Return lowest cost set of gains
- Plot Nichols and step response for verification
- Fine-tune by hand

```
J = (max(0,PM_des-PM)/PM_des)^2 + (max(0,GM_des-GMdB)/GM_des)^2 + (max(0, Ts5 - Ts5_des) / Ts5_des)^2;
if J < best
  best = J; bestG = [Kp Ki Kd]; bestL = Lo; bestT = T; bestStats = struct('PM',PM,'GMdB',GMdB,'Ts5',Ts5,'wgc_Hz',wgc_Hz);
end
```

Previous vs New Implementation (Current Gains)

No Delay				
Implementation	**GM (dB)**	**PM (deg)**	**Ts (sec)**	**wgc (Hz)**
Previous	12.2	43.8		0.156
New	14.6	38.3	10.1	0.27

3 Cycles of Delay				
Implementation	**GM (dB)**	**PM (deg)**	**Ts (sec)**	**wgc (Hz)**
Previous	8.31	27		0.156
New	7.4	35.6	9.4	0.27

Retuning of New Implementation

No Delay						
Kp	**Ki**	**Kd**	**GM (dB)**	**PM (deg)**	**Ts (sec)**	**wgc (Hz)**
0.5	0.1	0.95	14.6	38.3	10.1	0.27
0.75	0.133	1.183	13.3	44.2	10	0.313

3 Cycles of Delay						
Kp	**Ki**	**Kd**	**GM (dB)**	**PM (deg)**	**Ts (sec)**	**wgc (Hz)**
0.5	0.1	0.95	7.4	35.6	9.4	0.195
0.361	0.01	0.75	9.6	50.8	8.7	0.186
0.694	0.02	1.333	7.4	63	8.2	0.258
0.625	0.02	1.172	7.7	58.9	9.9	0.239

Next Steps

- Decide which implementation to use
- Finalize tuning in MATLAB
- Test in MAX