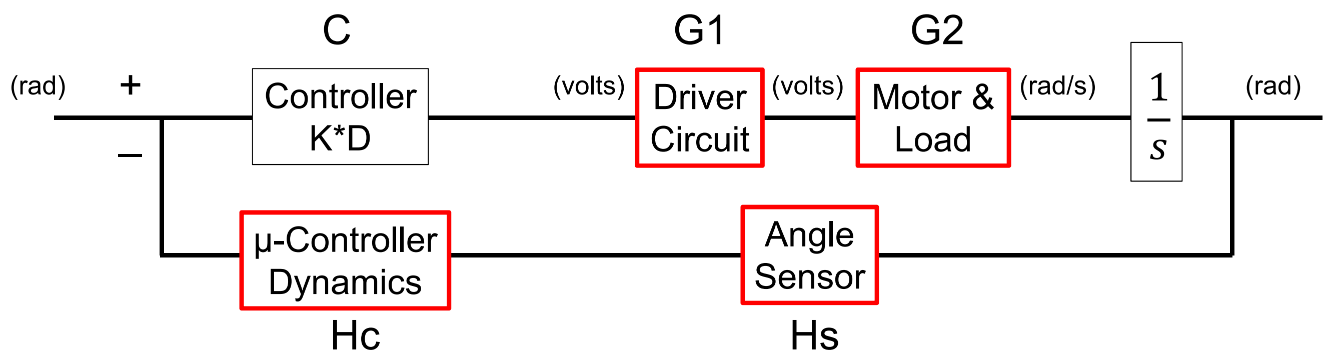
PD ~ PI ~ PID Controller Design

10-Step Process

(with PID Position-Control Example)

Prof. Leo Stocco, P.Eng
Electrical & Computer Engineering
University of British Columbia

Step 1 : System Identification



Develop Sub-System Models

- Black Box (experimental data curve)
    - APPROXIMATE as 2^{nd} order system
- White Box (full information)
    - Calculate from Data Sheet (**DS**) information
    - Reduce all internal loops to effective transfer function
- μ-Controller Dynamics (sample rate of DAQ = Control Frequency (**CF**))
    - Unity Gain filter with pole at 2CF

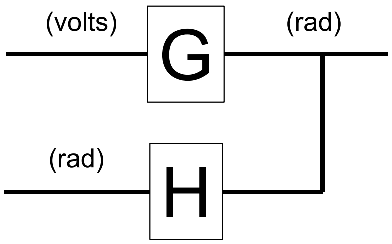
Compute Forward and Feedback Path Gains

- G = G1 * G2 * 1/s
- H = Hs * Hc (must have **UNITY DC GAIN**)
    - Optional: Design digital filter (Hc)

Linearize

- Neglect non-linearities (discontinuities, noise, etc.)

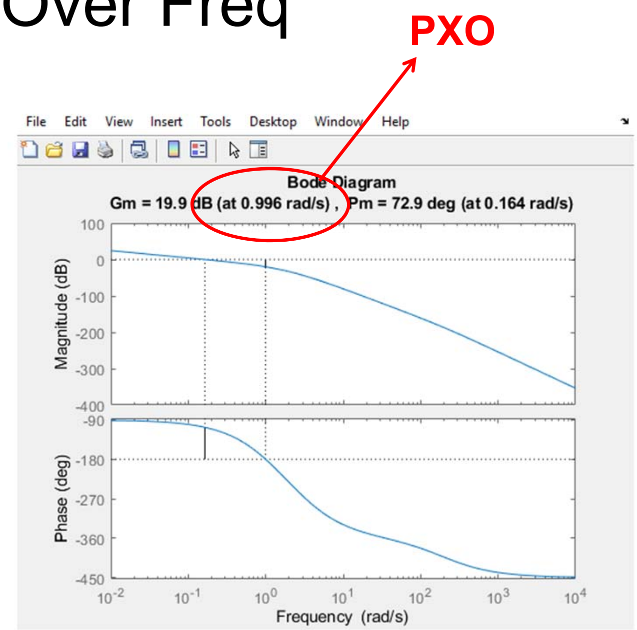
Step 2 : Find Phase X-Over Freq



$$G = \frac{1}{s(s+1)(s+2)(s+3)}$$

$$H = \frac{2CF}{(s+2CF)}$$

$$z = \frac{\omega_X}{10} = 0.1$$



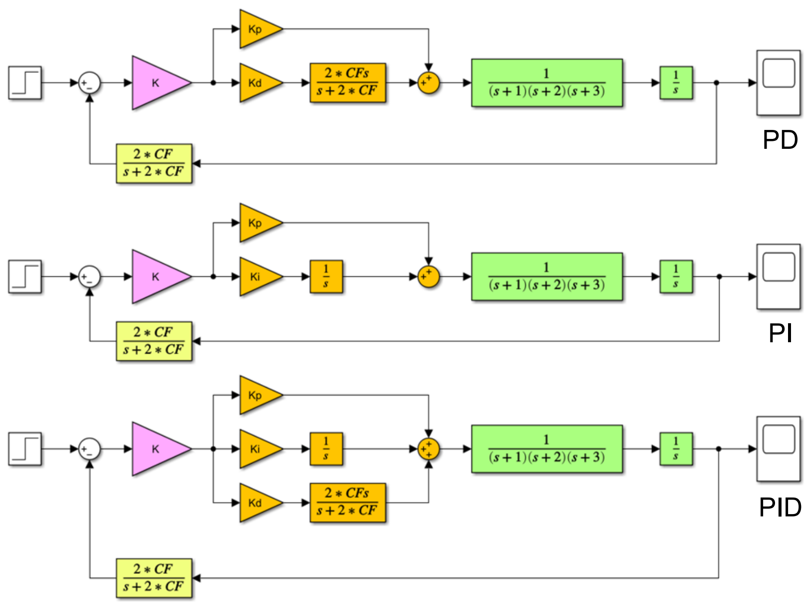
<u>Find Initial Zero Location</u>

- **margin()** gives Gain & Phase X-over frequencies (GXO & PXO)
- Initial zero 1 decade before PXO

<u>Example:</u>

- PXO = 1
- z = 0.1
- initial controller zero @ -0.1

Step 3 : Compute Controller Dynamics



$$D = K_p + K_d \frac{2CFs}{s + 2CF}$$

$$D = K_p + K_i \frac{1}{s}$$

$$D = K_p + K_i \frac{1}{s} + K_d \frac{2CFs}{s + 2CF}$$

Compute Micro-Controller Dynamics

- p = 2CF

PD Controller (zero @ -z)

- Kp = 1
- Kd = 1/z – 1/p

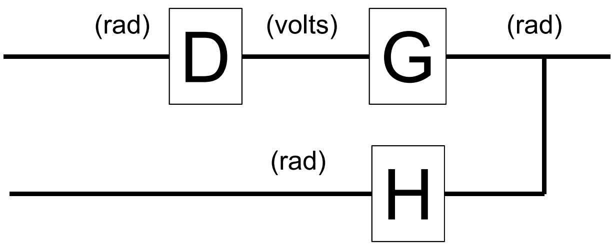
PI Controller (zero @ -z)

- Kp = 1/z
- Ki = 1

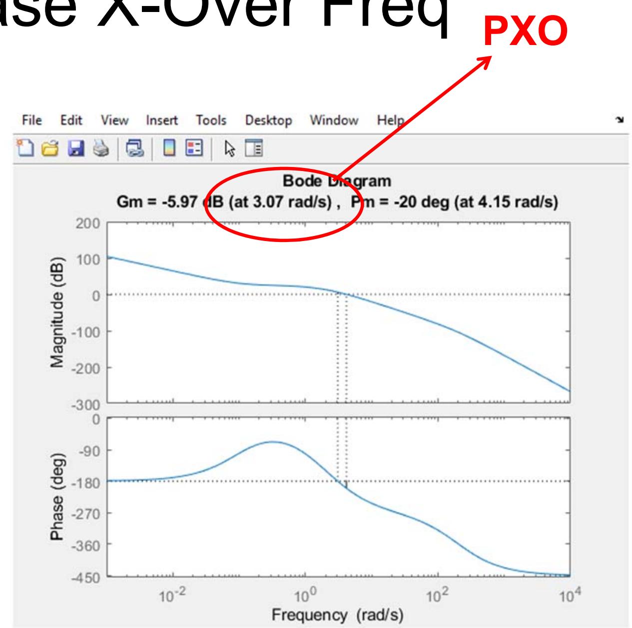
PID Controller (double-zero @ -z)

- Kp = 2/z – 1/p
- Ki = 1
- Kd = 1/z^2 – Kp/p

Step 4 : Find New Phase X-Over Freq



margin(D*G*H)



Find New Zero Location

- Run **margin()** with controller dynamics included
- New zero 1 decade before Phase X-Over frequency PXO

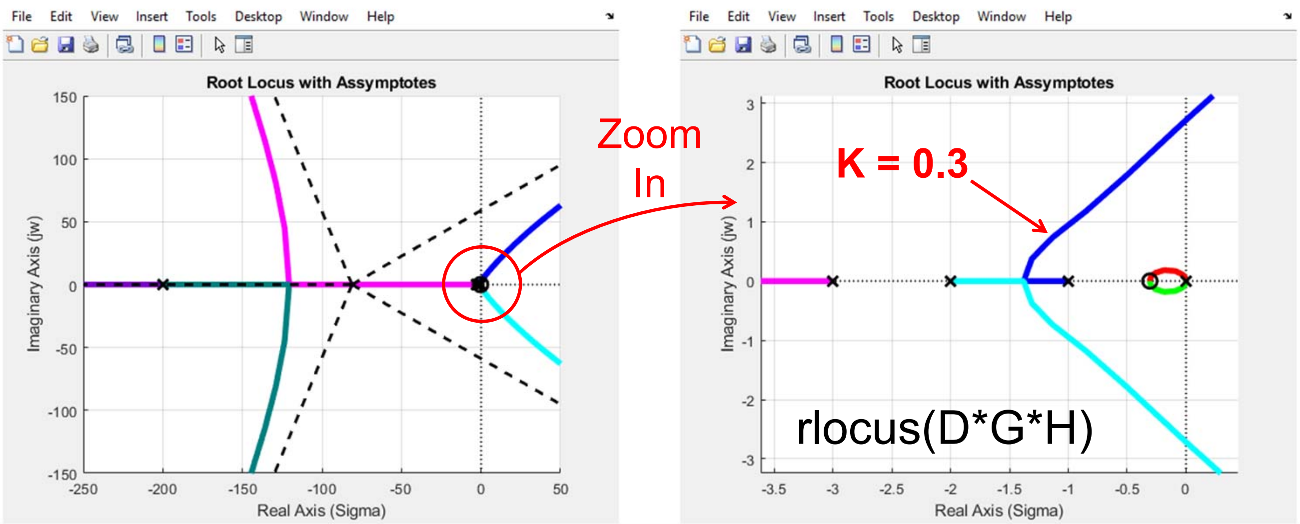
Iterate

- Re-Compute Dynamics (Step 3)
- Find new Zero (Step 4)
- Repeat until Zero stops changing

Multiple Solutions

- When phase = -180 at >1 frequency you get multiple solutions
- Choose lowest frequency solution (closest to jω axis on pole-zero plot)

Step 5 : Initial Gain K



Rule-Of-Thumb : Place Zero(s) at GXO

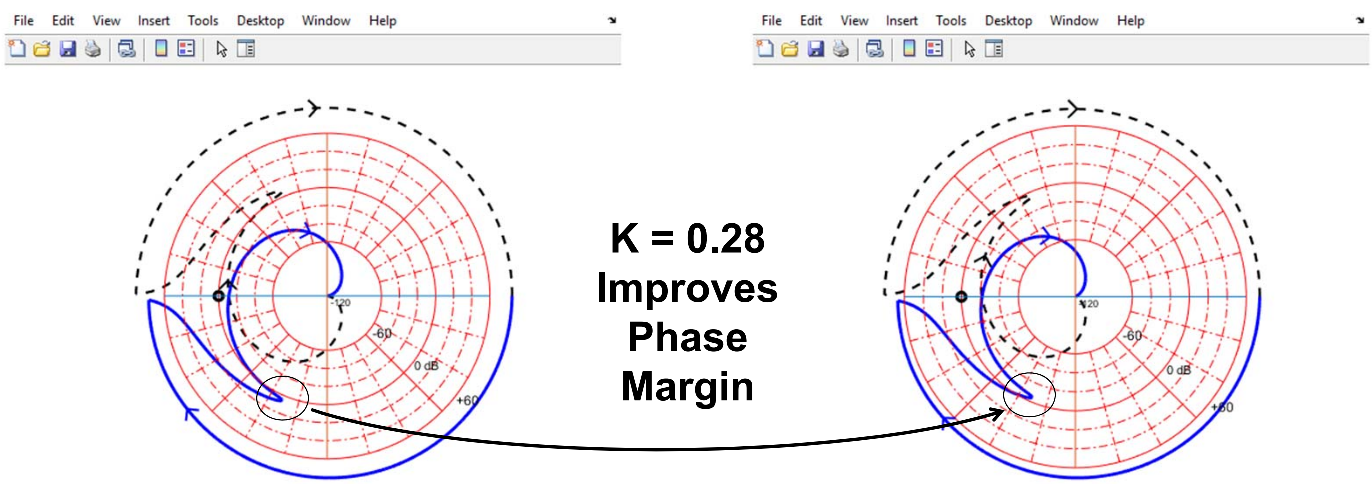
- Use **K** to place zero(s) at Gain X-Over Frequency (GXO)
    - **K = 1/abs(freqresp(DGH,z))**
    - GXO 1 decade below PXO
    - GXO < PXO → Stable (usually)
    - GXO = PXO → Marginally Stable (always)

Compute Root Locus (DGH)

- Zoom in on Dominant Roots
    - 1+ Open-Loop poles at zero?
        - 1 Controller root + any System roots
    - Controller zeros as expected?
        - Near jω axis to attract poles @ 0
    - De-stabilizing poles near real axis & far from imaginary axis?
- Adjust zeros if needed
    - **R-O-T** not always effective

Step 6 : Check Result

nyqlog(D*G*H) nyqlog(**K***D*G*H)



<u>Compute Nyquist Contours</u>

- DGH
- KDGH

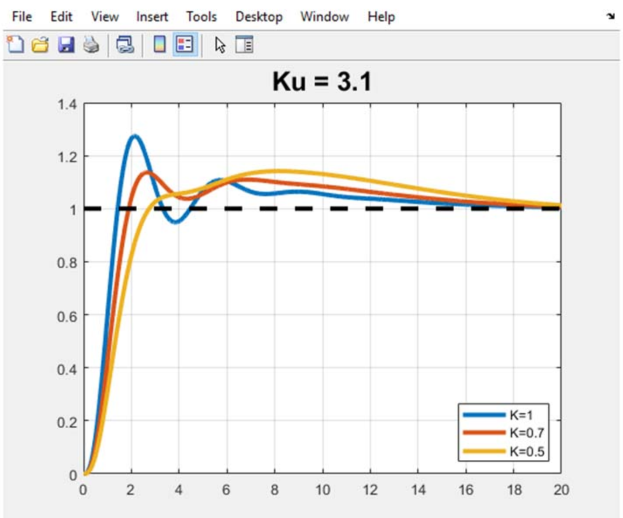
<u>Before Applying Gain K</u>

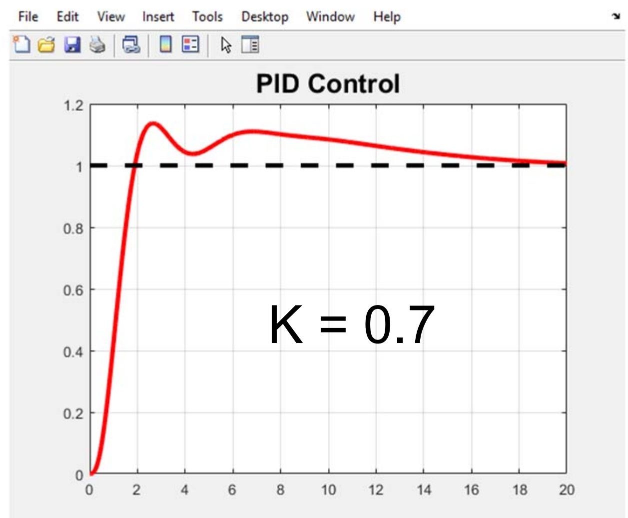
- Phase change reversal from controller zero(s)
- Corner has large phase margin (far from 180°)
- If not, repeat Step 4 & adjust zeros.

<u>After Applying Gain K</u>

- Corner nearer to 0dB iso-line
- Phase margin improved
- If not, repeat step 5 & check for errors

Step 7 : Step Response





Plot Step Response

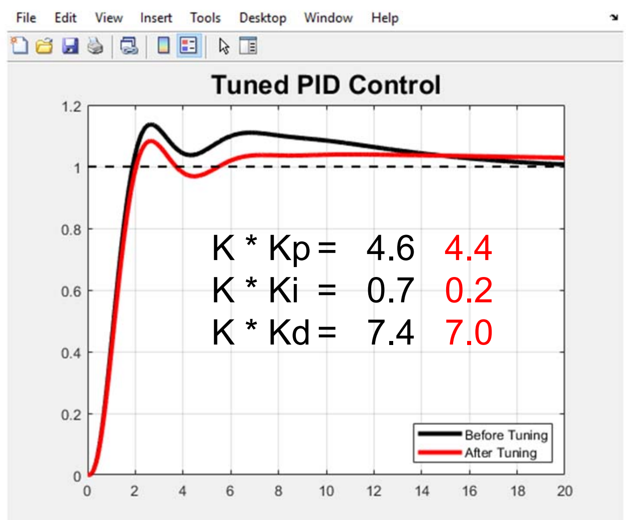
- Compute Closed-Loop transfer function
- Plot step response for range of K values

Choose best compromise

- Consult RCGs

Step 8a : Heuristic Tune

- Controller Gain K ↑
  ↑ Kp, Ki, Kd Simultaneously
  Poles follow Root Locus
- Proportional Gain Kp ↑
  ↓ Rise Time & Steady-State Error
  ↑ Overshoot
- Integral Gain Ki ↑
  ↓ Steady-State Error
  ↑ Overshoot, Settle Time
- Derivative Gain Kd ↑
  ↓ Overshoot, Settle Time
    - Destabilizes when too large
    - Depends on filter pole



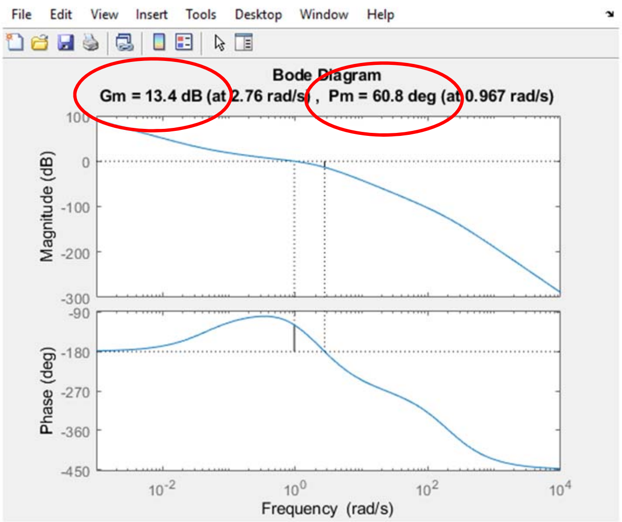
Adjust Individual Gains

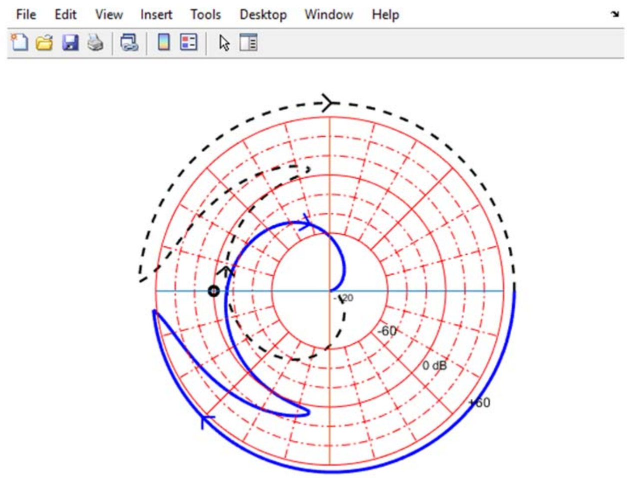
- #1 – Ki & K
    - Balance overshoot & steady-state error
- #2 – Kp & K
    - Balance rise time & stability
- #3 – Kd & K
    - Maximize stability

Repeat until satisfied

- Small increments
- Keep track of good combinations

Step 8b : Evaluate Margins





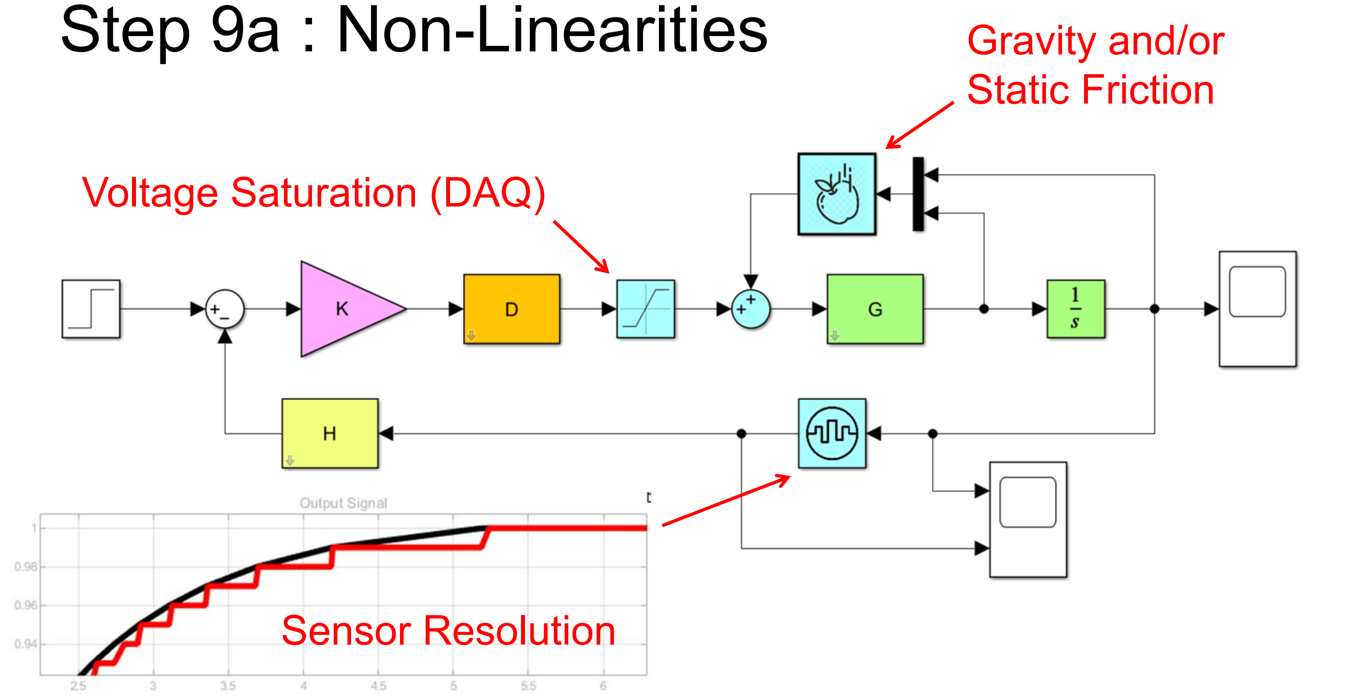
<u>Generate Bode & Nyquist plots</u>

- Evaluate Gain Margin Gm
- Evaluate Phase Margin Pm

<u>Check</u>

- Higher margins → Reduced sensitivity

Step 9a : Non-Linearities



Transfer to Simulink

- **Control System Toolbox / LTI System** block for transfer functions
- Non-linearities convenient to model in Simulink

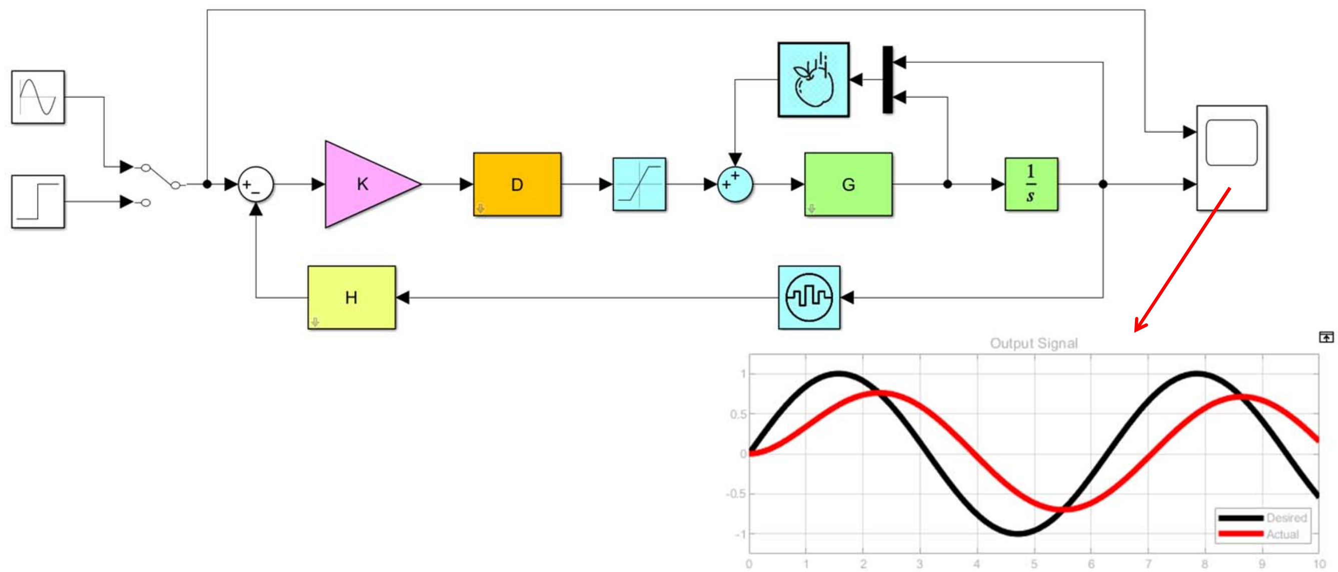
Add Non-Linearities

- **Discontinuities / Saturation** for Voltage / Current limits
- **Math Operations / Floor** for resolution
- **User Defined Functions / MATLAB function** for custom equations (Gravity / Friction)
- Explore all Simulink libraries for other features

Go To Step 7

- Adjust RCGs

Step 9b : Moving Target



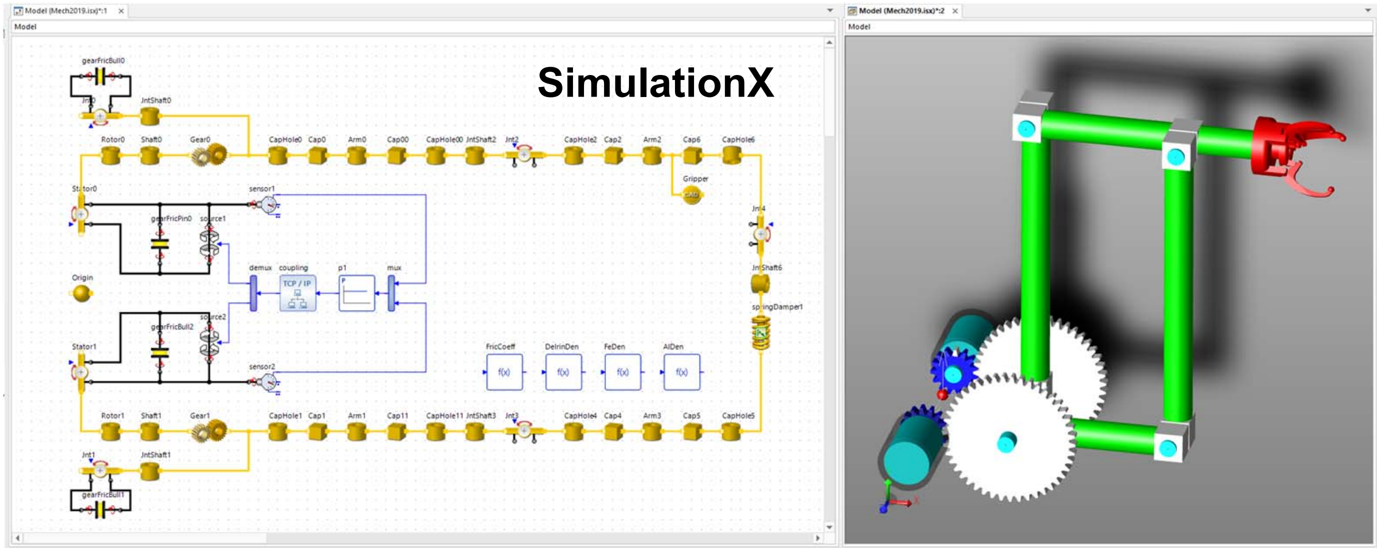
Replace Step Input with Sine Input

- Evaluate Delay
- Overshoot eliminated by moving target
- Better tracking when **STABILTY REDUCED**

Go To Step 7

- Adjust RCGs

Step 10 : Intended System



Apply Controller to Real or Simulated System

- Results similar?
  - Fix bugs and repeat process.

- Results acceptable?
  - Repeat Heuristic tuning on intended system