

Brief

Software implementation of the parallel PID controller seen in Figure 1. Written in C, for use in embedded systems equipped with a floating-point engine.

Features:

- Derivative low-pass-filter (LPF) with configurable pole location
- Configurable controller output saturation
- Configurable anti-windup functionality

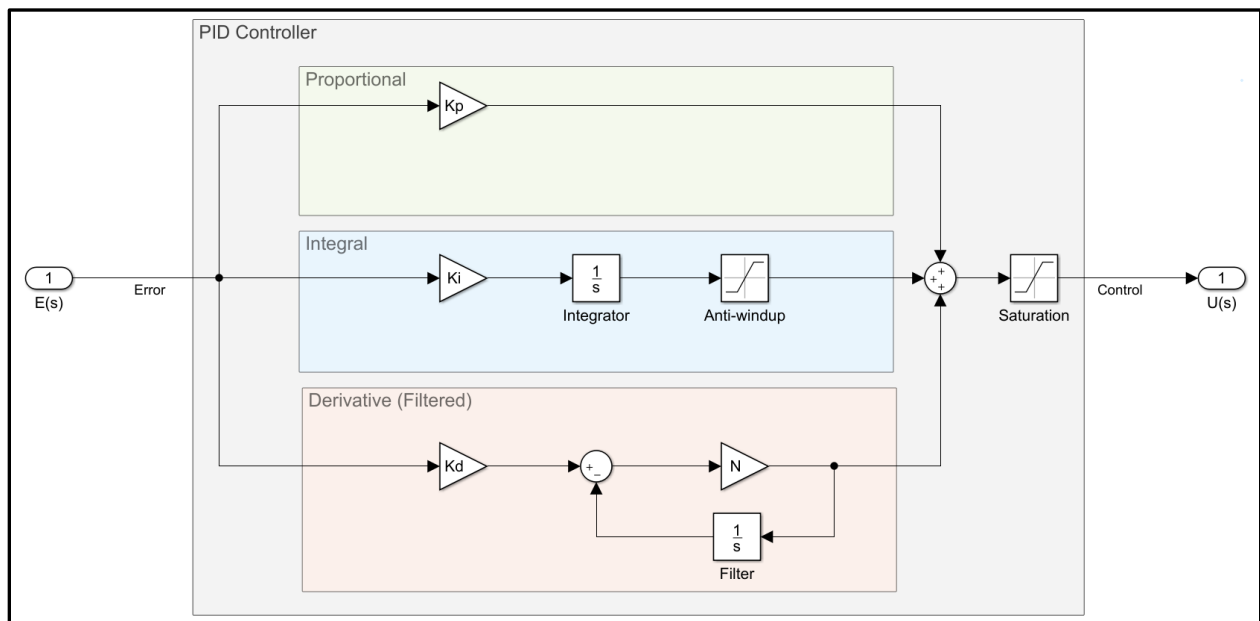


Figure 1: Implemented Parallel PID Controller

Theory

We start with the s-domain transfer function of the PID controller seen in Figure 1:

Equation 1: PID s-domain transfer function

$$C(s) = \frac{U(s)}{E(s)} = K_p + \frac{K_i}{s} + K_d \frac{Ns}{s + N}$$

The PID transfer function, $C(s)$, is discretized using the Bilinear (Tustin) transform:

Equation 2: Bilinear (Tustin) Transform

$$s = \frac{2(z-1)}{T(z+1)}$$

Yielding the z-domain transfer function, $C(z)$:

Equation 3: PID z-domain transfer function

$$C(z) = \frac{U(z)}{E(z)} = \frac{N(z)}{D(z)} = \frac{K_p\alpha_1 + K_i\alpha_2 + K_d\alpha_3}{(2NT + 4)z^2 - 8z + (4 - 2NT)}$$

Where:

$$\begin{aligned}\alpha_1 &= (4 - 2NT)z^2 - 8z + (4 - 2NT) \\ \alpha_2 &= (NT^2 + 2T)z^2 + (2NT^2)z + (NT^2 - 2T) \\ \alpha_3 &= 4Nz^2 - 8Nz + 4N\end{aligned}$$

Cross multiplying then solving for the highest-order coefficient operating on $U(z)$:

$$U(z)D(z) = E(z)N(z)$$

Equation 4: Controller output z-domain equation

$$\begin{aligned}
 U(z) = & K_p \left[E(z) - E(z)z^{-1} \left(\frac{8}{\beta_1} \right) - E(z)z^{-2} \left(\frac{2NT - 4}{\beta_1} \right) \right] \\
 & + K_i \left[E(z) \left(\frac{NT^2 + 2T}{\beta_1} \right) + E(z)z^{-1} \left(\frac{2NT^2}{\beta_1} \right) \right. \\
 & \left. - E(z)z^{-2} \left(\frac{2T - NT^2}{\beta_1} \right) \right] \\
 & + K_d \left[E(z) \frac{4N}{\beta_1} - E(z)z^{-1} \frac{8N}{\beta_1} + E(z)z^{-2} \frac{4N}{\beta_1} \right] \\
 & + U(z)z^{-1} \left(\frac{8}{\beta_1} \right) + U(z)z^{-2} \left(\frac{2NT - 4}{\beta_1} \right)
 \end{aligned}$$

The time-shifting property is used to convert the z-domain equation, $U(z)$, to the sampled-time equation $U(n)$:

Equation 5: Time-shifting property

$$z^{-t}X(z) = x(n - t)$$

Equation 6: Controller output sampled-time equation

$$\begin{aligned}
 U(n) = & K_p \left[E(n) - E(n-1) \left(\frac{8}{\beta_1} \right) + E(n-2) \left(\frac{4 - 2NT}{\beta_1} \right) \right] \\
 & + K_i \left[E(n) \left(\frac{T}{2} \right) + E(n-1) \left(\frac{NT}{N + \frac{T}{2}} \right) + E(n-2) \frac{T}{2} \left(\frac{N - \frac{2}{T}}{N + \frac{T}{2}} \right) \right] \\
 & + K_d \left[E(n) \frac{4N}{\beta_1} - E(n-1) \frac{8N}{\beta_1} + E(n-2) \frac{4N}{\beta_1} \right] + U(n-1) \left(\frac{8}{\beta_1} \right) \\
 & + U(n-2) \left(\frac{2NT - 4}{\beta_1} \right)
 \end{aligned}$$

Where:

$$U(n) = \text{current controller output}$$

$U(n - 1) = \text{output } T \text{ second}(s) \text{ ago}$

$U(n - 2) = \text{output } 2T \text{ second}(s) \text{ ago}$

$E(n) = \text{current error (controller input)}$

$E(n - 1) = \text{error } T \text{ second}(s) \text{ ago}$

$E(n - 2) = \text{error } 2T \text{ second}(s) \text{ ago}$

$Kp = \text{proportional gain}$

$Ki = \text{integral gain}$

$Kd = \text{derivative gain}$

$N = \text{derivative low pass filter pole location } \left(\frac{\text{rad}}{s} \right)$

$T = \text{sample time (seconds)}$

$\beta_1 = 2NT + 4$

Equation 6 is the [infinite impulse response \(IIR\)](#) of the PID controller in Figure 1. The PID exhibits an infinite impulse response (as opposed to a finite impulse response) due to its dependence on past outputs, introduced by the filter feedback path implemented in the derivate branch of the controller. The difference equation is implemented in C.

Validation

Equation 6 was implemented in a MATLAB Simulink model, controlling a stable plant with a transfer function of $1/(s+1)$. The step response of Equation 6 was compared to that of the built-in, MATLAB discrete-time parallel PID block whose gain parameters were equivalent and driving an identical plant. The results can be seen below.

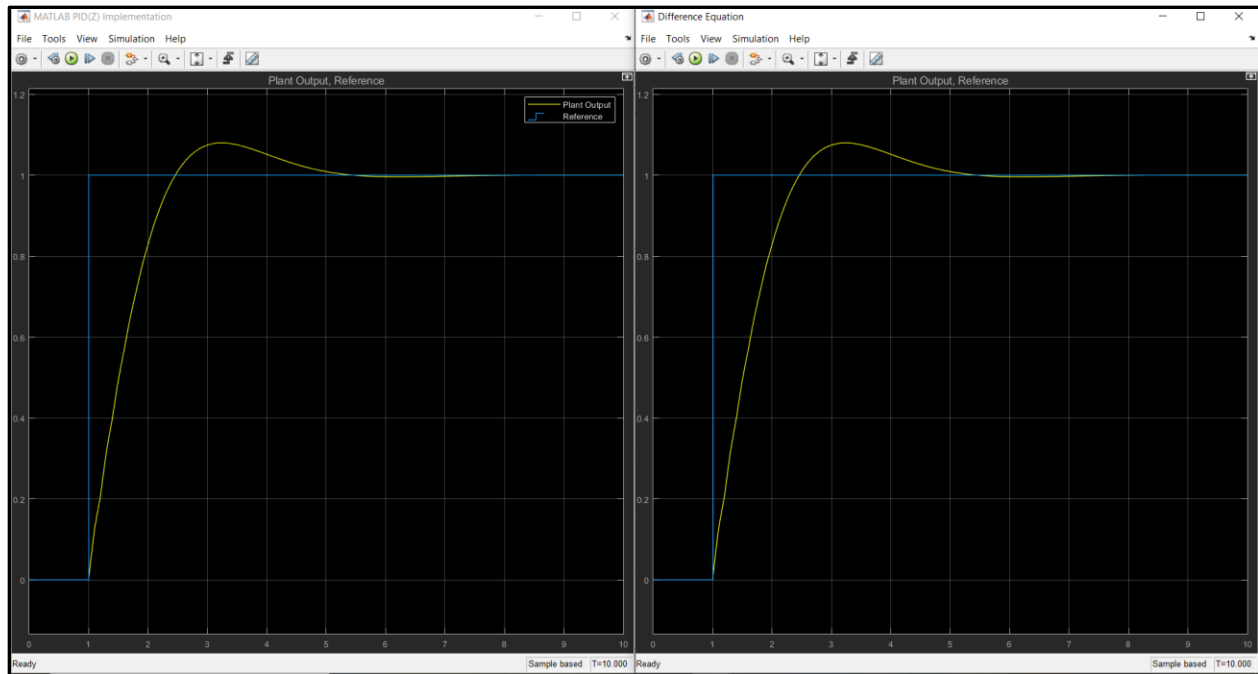


Figure 2: Step response of closed loop system. System parameters: $K_p = 1$, $K_i = 2$, $K_d = 0.0125$, $T = 100\text{ms}$ (10 Hz), $N = 20\pi$ rad/s (20 Hz). Left is MATLAB PID(z) block. Right is Equation 6 implementation.

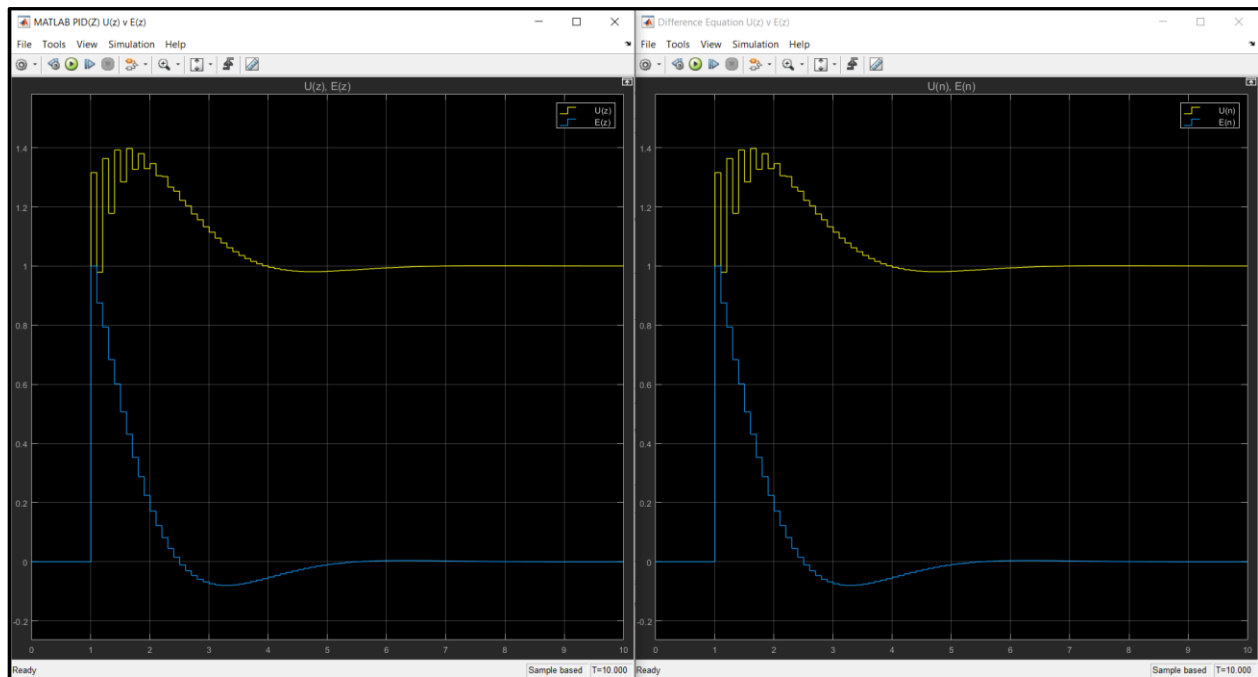


Figure 3: Controller output, $U(z)$, v controller input, $E(z)$. System parameters: $K_p = 1$, $K_i = 2$, $K_d = 0.0125$, $T = 100\text{ms}$ (10 Hz), $N = 20\pi$ rad/s (20 Hz). Left is MATLAB PID(z) block. Right is Equation 6 implementation.

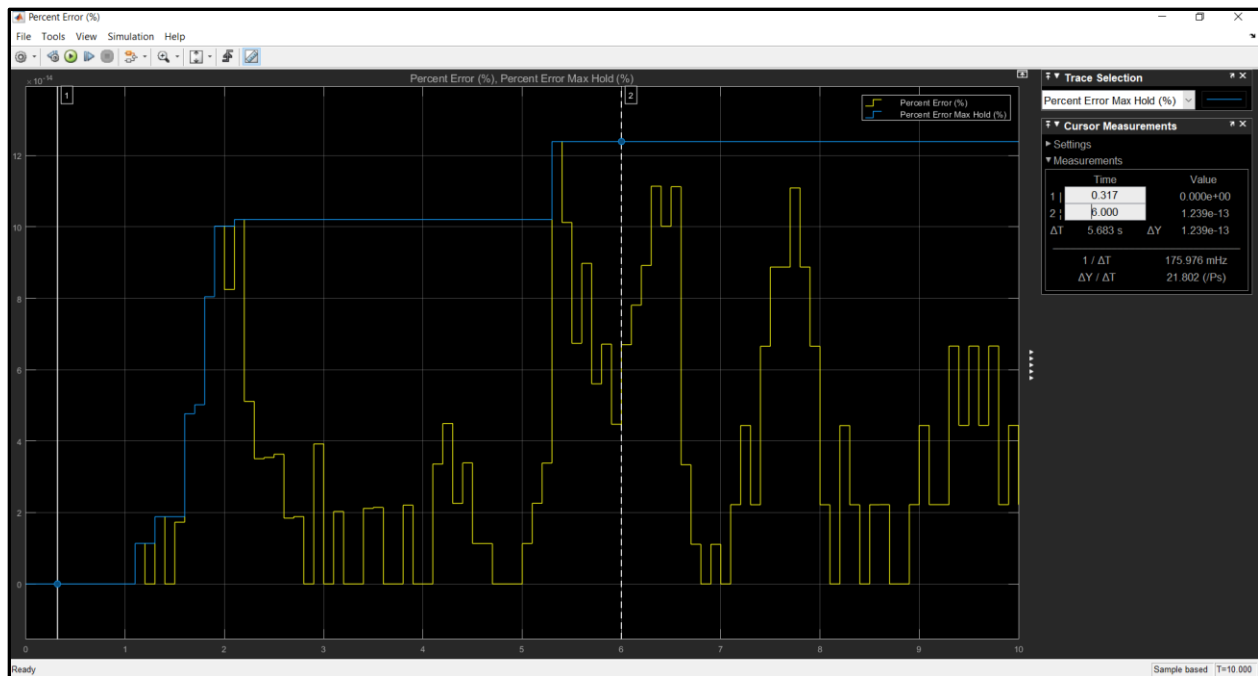


Figure 4: Percent error between expected controller output (MATLAB PID(z) block) and the difference equation output. Note that the max error throughout the 10 second simulation is 1.239e-13 %.