PID 11082

Closed Loop Control on a 16F PICmicro

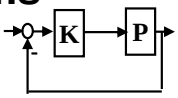
Class Objective

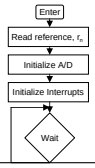
When you finish this class you will:

- Define the standard Feedback Loop and the "Big Picture" of controls
- Describe the steps needed to produce and analyze a simple system transfer function model
- Explain a PID controller's ability to stabilize or enhance a systems performance
- Label the components of the Signal Chain Block Diagram for a controlled feedback system
- Describe each block of code needed for a PID controller in a 16F PICmicro

Agenda

- **Introduction to Control Systems**
- **Modeling**
  - Examples
- **Feedback Control Systems**
  - Examples
- **PID Implementation**
  - LAB

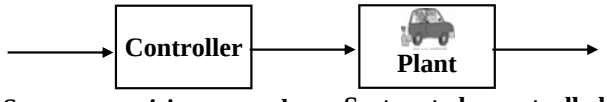




Introduction

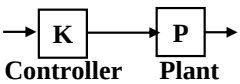
What is a Control System?

- A **control system** is an arrangement of physical components connected or related in such a manner as to command, direct, or regulate itself or another system



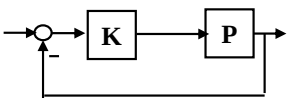
Open vs Closed Loop

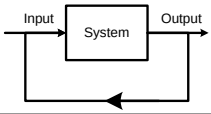
- Open Loop Control
  - Control action independent of output
  - Lots of a priori info about plant behavior needed
  - Accuracy depends on calibration
  - Suffers from noise and uncertainty



Open vs Closed Loop

- Closed Loop Control
  - Requires Feedback form system output
  - Control action depends on the error e=r-y
  - We use feedback to predictably enhance/shape performance or alter undesirable properties in the presence of uncertainty and disturbances
  - Increase accuracy, Decrease sensitivity
  - Instability can be an issue





Why do we need models?

- Mathematical models are needed for quantative relationships
- Mathematical models are essential for
  - Designing high performance control systems
  - Conceptualization
  - Simulation/Analysis
  - Prototyping
  - Validation
  - Deployment

Modeling

Modeling

Time Domain

Schematic

$$\sum F(t) = m \cdot a(t)$$

$V(t)$ → Dynamic Equation $\dot{y}(t) + y(t) = u(t)$ → $\dot{\theta}(t)$

Analysis

Step Response

Frequency Domain

Laplace Transform

$$s \cdot Y(s) + Y(s) = U(s)$$

Transfer Function

$$P(s) = \frac{1}{s+1}$$

Analysis

pole-zero

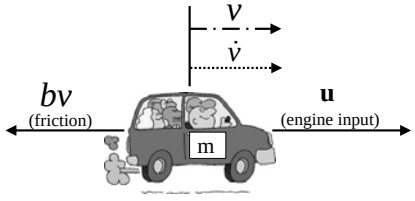
Frequency Response

Schematic & Dynamic Equation

- **Schematic**
  - Identify system to be controlled
  - Form Free body diagram
  - Label inputs and outputs
- **Dynamic equation**
  - Write the equation of the free body diagram from fundamental principles

Schematic & Dynamic Equation

Example Free body diagram:



Dynamic Equation:

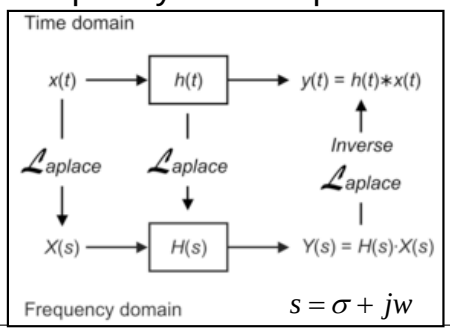
Newton's Law: $F = ma$

$$m\dot{v}(t) = u(t) - bv(t)$$

$$y(t) = v(t)$$

Laplace Transforms

- **Used to analyze linear systems**
  - Reduces mathematical complexity
  - New Frequency Domain point of view



Laplace Transforms

$$F(s) = \mathcal{L}\{f(t)\} = \int_{0^-}^{\infty} e^{-st} f(t)\, dt$$

- **Common Transform pairs:**

$$1 \quad \leftrightarrow \quad \frac{1}{s} \qquad\qquad \cos\omega t \quad \leftrightarrow \quad \frac{s}{s^2+\omega^2} = \frac{1/2}{s-j\omega} + \frac{1/2}{s+j\omega}$$

$$e^{at} \quad \leftrightarrow \quad \frac{1}{s-a} \qquad\qquad \sin\omega t \quad \leftrightarrow \quad \frac{\omega}{s^2+\omega^2} = \frac{1/2j}{s-j\omega} - \frac{1/2j}{s+j\omega}$$

Laplace Transforms

- **Common Transform properties**

Linearity $af(t) + bg(t) \Leftrightarrow aF(s) + bG(s)$

Differentiation $f'(t) \Leftrightarrow sF(s) - f(0^-)$

Integration $\int_0^t f(t)dt \Leftrightarrow \frac{1}{s}F(s)$

Laplace Transforms

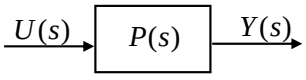
- **Car example**
  - Remember: $f'(t) \Leftrightarrow sF(s) - f(0^-)$

$$m\dot{v}(t) = u(t) - bv(t) \quad \Rightarrow \quad msV(s) = U(s) - bV(s)$$
$$y(t) = v(t) \qquad\qquad Y(s) = V(s)$$

The Transfer Function

- **The Transfer Function is the linear mapping of the input, $U(s)$, to the output $Y(s)$**

$$P(s) = \frac{Y(s)}{U(s)}$$

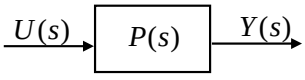


The Transfer Function

- **Ratio of Output over Input for car example**

$$msV(s) = U(s) - bV(s)$$
$$Y(s) = V(s)$$

$$\frac{Y(s)}{U(s)} = \frac{V(s)}{msV(s) + bV(s)} = \frac{1/m}{s + b/m} = P(s)$$



Pole/Zero Stability

$$P(s) = \frac{N(s)}{D(s)}$$

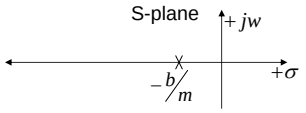
- **Zeros are the roots of N(s)**
  - Represented on the s-plane as a circle
  - Transfer function's magnitude at this point is zero $|P(z_0)| = 0$

Pole/Zero Stability

$$P(s) = \frac{N(s)}{D(s)}$$

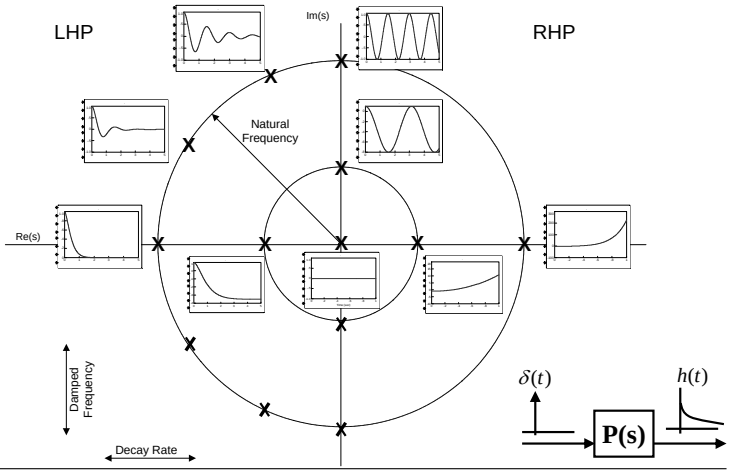
- **Poles are the roots of D(s)**
  - Represented as an x on the s-plane
  - Transfer function's magnitude at this point is infinity $|P(p_0)| = \infty$

$$P(s) = \frac{1/m}{s + b/m}$$



Pole/Zero Stability

Impulse Responses:

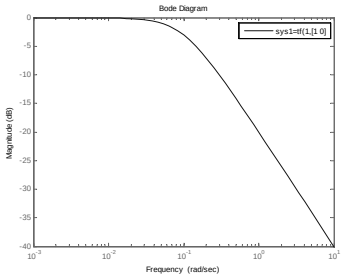


Frequency Response

- **Set s=jw, P(jw)**
  - Calculate Magnitude Response |P(jw)|
  - And Phase Response Angle(P(jw))

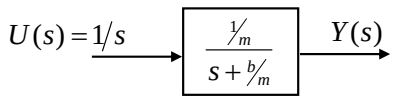
$$P(jw) = \frac{1/m}{jw + b/m}$$

$$|P(jw)| = \frac{1/m}{\sqrt{w^2 + (b/m)^2}}$$



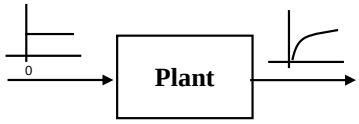
Step Response

- **The response of the system to a unit step input**



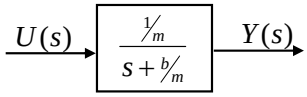
$u(t) = 1 \quad at\ t = 0$

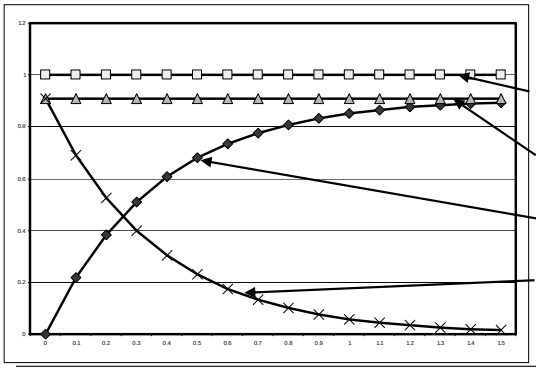
$$y(t) = L^{-1}\{Y(s)\} = L^{-1}\left\{\frac{1/m}{s(s + b/m)}\right\}$$



Step Response

- **Step response**





Unit Step

1/b max velocity

y(t) step response

g(t) impulse response

Interpreting the Car Model

- **Reality Check, does it make sense?**
- **Is it stable? Why**
- **Note:**
  - As friction (b) increases
  - The steady state speed or "terminal velocity" (1/b) decreases

Model Limitations

- **"All models have limitations, stupidity does not."**

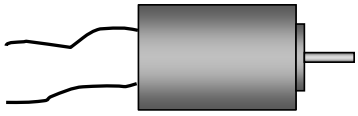
  – A. Rodriguez

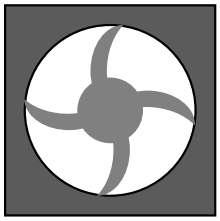
Car Simulation Lab

- **Lets test this model**

Motor Example

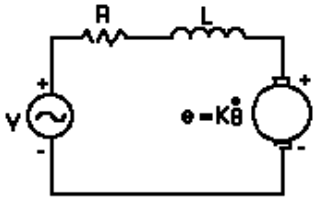
- **DC Motor**

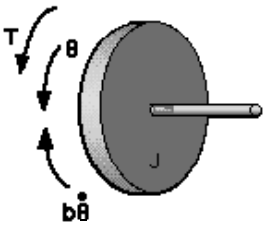




  - Motor speed is proportional to voltage $V \propto K_1\dot{\theta}$
  - Motor torque is proportional to current $T \propto K_2 i$

Motor's Schematic and Dynamic Equations





- **Sum of voltages:** $L\frac{di}{dt} + Ri = V - K_1\dot{\theta}$
- **Sum of torques:** $J\ddot{\theta} + b\dot{\theta} = K_2 i$
- **Input:** V **Output:** $w = \dot{\theta}$

Motor's Transfer Function

- **Remember:** $f'(t) \Leftrightarrow sF(s) - f(0^-)$

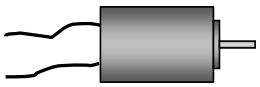
$$L\frac{di}{dt} + Ri = V - K_1\dot{\theta} \qquad (Ls+R)I(s) = V - K_1 W(s)$$
$$J\ddot{\theta} + b\dot{\theta} = K_2 i \longrightarrow (Js+b)W(s) = K_2 I(s)$$

Input : Volatage V

Output : Angular velocity $w = \dot{\theta}$

- **Output over Input:** $\frac{W}{V} = \frac{K_2}{(Js+b)(Ls+R)+K_1^{\,2}}$

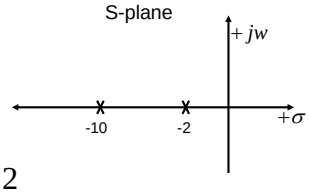
Motor's Transfer Function



$$P(s) = \frac{K_2}{(Js+b)(Ls+R)+K_1^{\,2}}$$

$$K_1 = K_2 = J = 0.01$$

$$L = 0.5,\ R = 1,\ b = 0.1$$

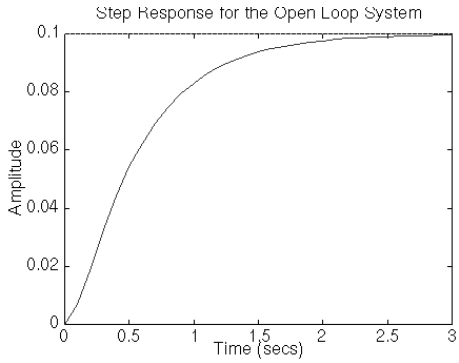


$$P(s) = \frac{2}{s^2+12s+20.02} \approx \frac{2}{(s+10)(s+2)}$$

- **Is it stable?**

Analysis

- **Step Response** $P(s) = \frac{2}{(s+10)(s+2)}$



Motor Simulation Lab

- **Lets test this model**

Modeling

Time - Domain

Schematic

$\sum V = 0$ $\sum T = 0$

$V(t)$ Dynamic Equation $\dot{\theta}(t)$

$$L\frac{di}{dt} + Ri = V - K_1\dot{\theta}$$

$$J\ddot{\theta} + b\dot{\theta} = K_2 i$$

Analysis

Step Response

S - Domain

Laplace Transform

$$(Ls+R)I(s) = V - K_1 W(s)$$

$$(Js+b)W(s) = K_2 I(s)$$

Transfer Function

$$\frac{W(s)}{V(s)} = P(s) = \frac{2}{(s+10)(s+2)}$$

Analysis

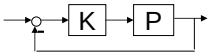
pole-zero

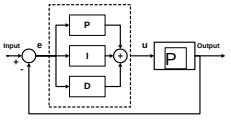
Frequency Response

Feedback Control

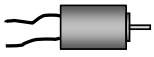
Feedback Control

- **Feedback**
  - The Standard Feedback Loop
  - Closed Loop



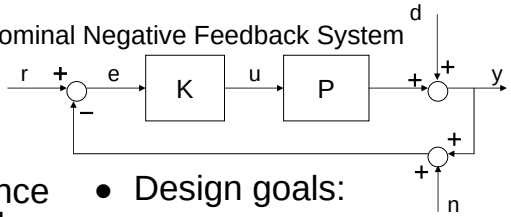


- **PID**
  - How does each term work?
  - Tuning a PID controller
- **Car example**
- **Motor example**



The Standard Feedback Loop and the "Big Picture"

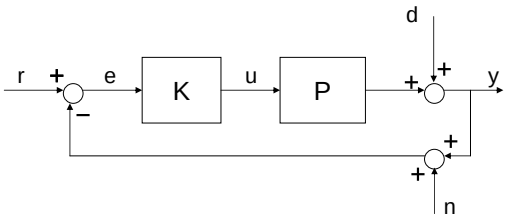
Nominal Negative Feedback System



- r – reference command
- e – error
- u – control
- d – disturbance
- y – output
- n – sensor noise

- Design goals:
  - Stable
  - Small error
  - Good command following
  - Good disturbance/noise attenuation

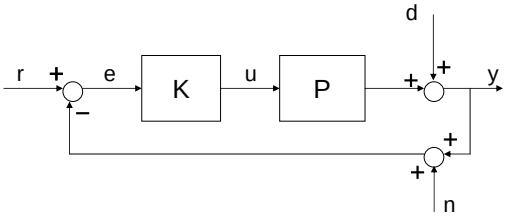
Closed Loop Transfer Function



$$y = PK(r - y) = PKr - PKy$$

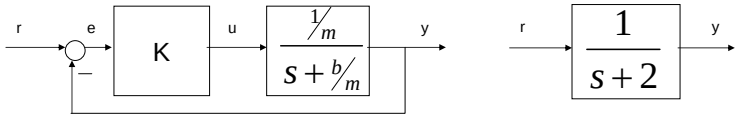
$$y = \frac{PKr}{1+PK} \qquad \frac{y}{r} = T_{ry} = \frac{PK}{1+PK}$$

Closed Loop Transfer Function



$$y = \underbrace{\left[\frac{PK}{1+PK}\right]}_{T_{ry}} r + \underbrace{\left[\frac{1}{1+PK}\right]}_{T_{dy}} d + \underbrace{\left[\frac{-PK}{1+PK}\right]}_{T_{ny}} n$$

Closed Loop Transfer Function



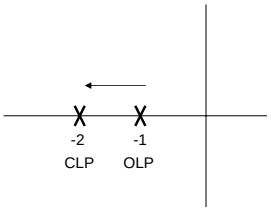
$$m = 1 \quad b = 1$$

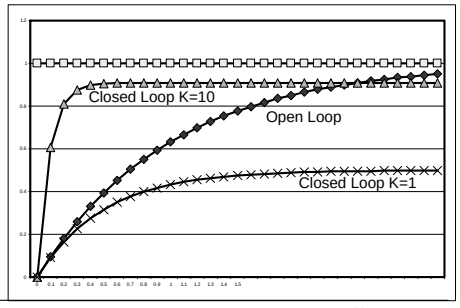
$$P = \frac{1}{s+1} \qquad K = 1$$

$$T_{ry} = \frac{PK}{1+PK} = \frac{1}{s+2}$$

Closed Loop Transfer Function

- Same analysis tools used for the closed loop transfer function
  - Pole-zero stability
  - Step response





PID

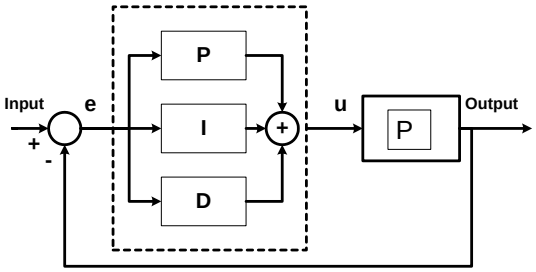
- The PID (Proportional Integral Derivative) is sometimes called a three term controller
  - Good robustness properties across a wide frequency range
  - Simple

$$\frac{U(s)}{E(s)} = k_p + \frac{k_i}{s} + k_d s$$

PID

- Three adjustable gains

$$u(t) = k_p e(t) + k_i \int_0^t e(t)dt + k_d \frac{de(t)}{dt}$$



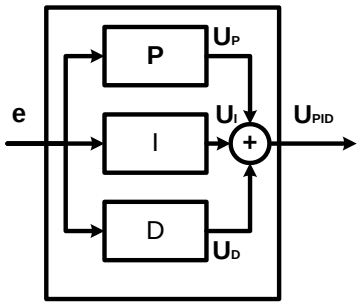
PID Control

Proportional -> Present

Integral -> Past

Derivative -> Future

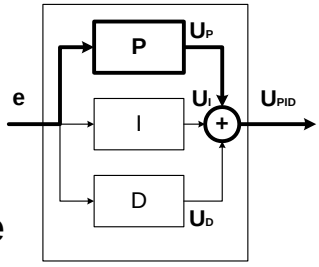
Today… P
Yesterday… I
Tomorrow… D



PID Control

Proportional

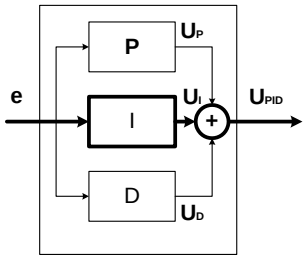
- Simple Gain
- $U_p = K_P * e$
- The larger the error, the larger the control effort
- **Decreases Rise Time**
- Increases Overshoot



PID Control

Integral

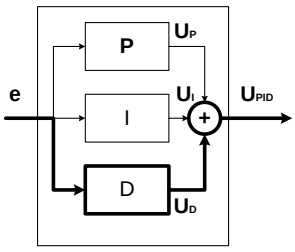
- "Averager"
- $U_I = K_I * \Sigma e$
- Low Pass Filter (*1/s*)
- Decreases Rise Time
- Increases Overshoot
- Adds sensitivity to low frequency input
- **Eliminates Steady State error to Step Inputs**



PID Control

Derivative

- Slope/Anticipator
- $U_D = K_D * de/dt$
- High Pass Filter (*s*)
- Increases Bandwidth
- **Decreases overshoot & settling time**
- Adds sensitivity to high frequency input components (and noise!)



PID Control

Parameter effects summary

Parameter	Rise Time	Overshoot	Settling Time	S.S. Error
P	Decrease	Increase	Small Change	Decrease
I	Decrease	Increase	Increase	Eliminate
D	Small Change	Decrease	Decrease	Small Change

In short:
P to decrease rise time
I to eliminate Steady State error
D to decrease overshoot

Tuning a PID controller

Practical, Closed loop tuning

1. Set **I** and **D** to zero.
2. Increase **P** until output oscillates.
3. Increase **I** until oscillation stops.
4. Increase **D** until the loop is acceptably quick to reach its reference

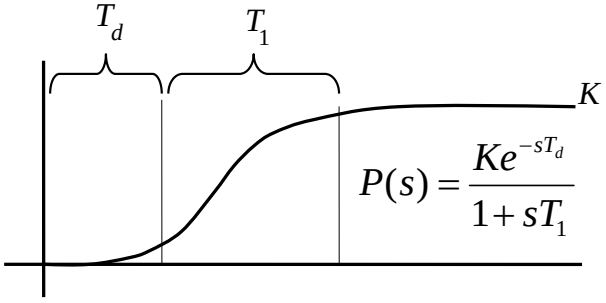
Tuning a PID controller

- Ziegler-Nichols closed loop tuning
  - Disable K_d and K_i
  - Perform step tests while increasing K_p until a stable oscillation is achieved. (This value of K_p is called the ultimate gain K_u)
  - Measure the period of the resulting oscillation and call it P_u

K	k_p	k_i	k_d
PI	$0.45K_u$	$1.2K_p/P_u$	
PID	$0.6K_u$	$2K_p/P_u$	$K_p*P_u/8$

Tuning a PID controller

- **Ziegler-Nichols open loop tuning**
  - Approximate your open loop step response with a delayed single order system:



Tuning a PID controller

- **Ziegler-Nichols open loop tuning**
  - Set your PID gains using this table and the measured values of T_1, T_d, and K

K	k_p	k_i	k_d
P	T_1/KT_d		
PI	$0.9T_1/KT_d$	$3.3T_d$	
PID	$1.2T_1/KT_d$	$2T_d$	$0.5T_d$

Car Example

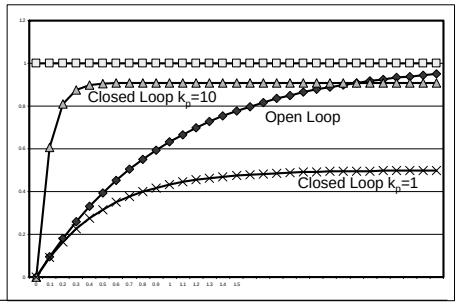
$$P(s) = \frac{1/m}{s + b/m}$$

- **K=k_p, Our First Controller**

$$m = 1 \quad b = 1$$

$$P = \frac{1}{s+1} \quad k_p = 1$$

$$T_{ry} = \frac{PK}{1+PK} = \frac{1}{s+2}$$



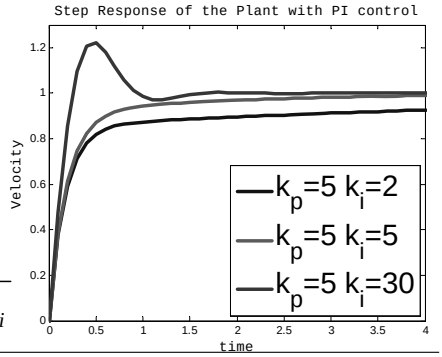
Car Example

$$P(s) = \frac{1/m}{s + b/m}$$

- **PI control**
- **e_{ss} is gone**

$$K = k_p + \frac{k_i}{s}$$

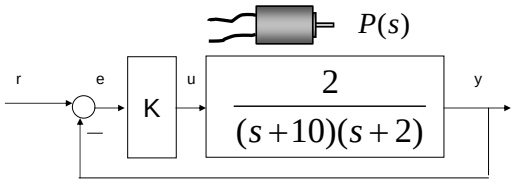
$$T_{ry} = \frac{k_p s + k_i}{s^2 + s(1+k_p) + k_i}$$



Car Simulation

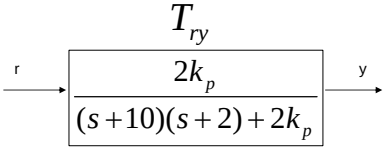
- **Car Simulation Lab**

Motor Example



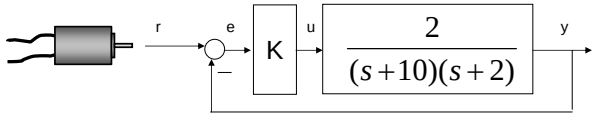
- **K=k_p, Our First Controller**

$$T_{ry} = \frac{PK}{1+PK}$$



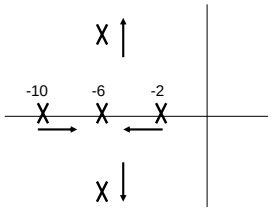
T_{ry}

Motor Example



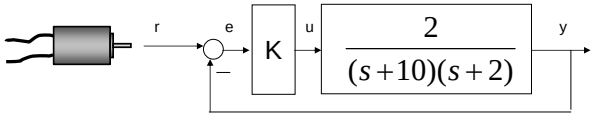
$$K = k_p$$

$$k_p = 8 \qquad T_{ry} = \frac{16}{(s+6)(s+6)}$$

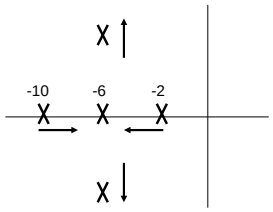


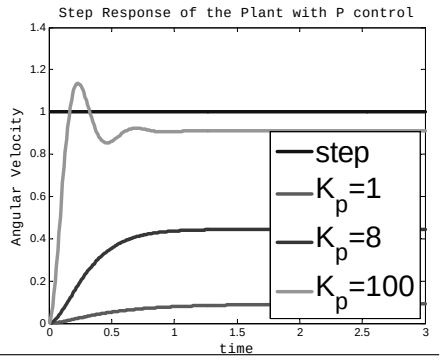
$$k_p = 100 \qquad T_{ry} = \frac{200}{(s+6+13.5j)(s+6-13.5j)}$$

Motor Example

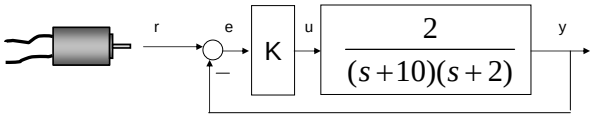


$$K = k_p$$



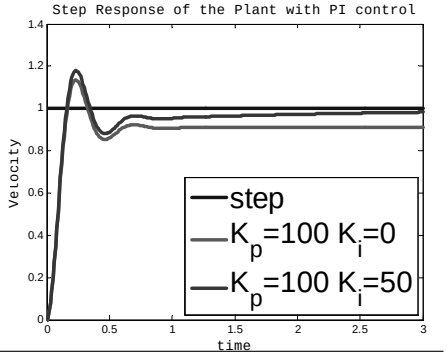


Motor Example

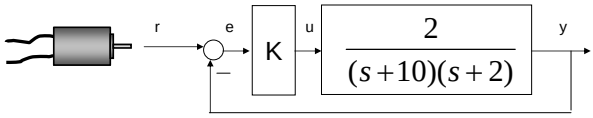


- **PI control**

$$K = k_p + \frac{k_i}{s}$$

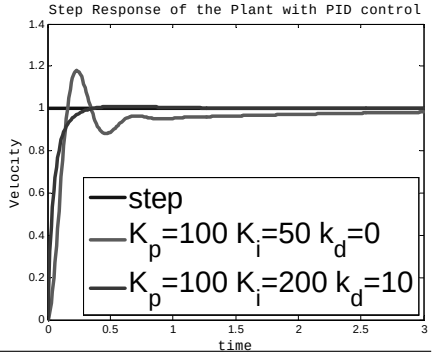


Motor Example



- **PID control**

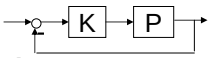
$$K = k_p + \frac{k_i}{s} + k_d s$$



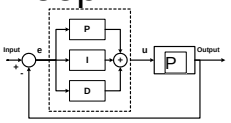
Motor Simulation

- **Motor Simulation Lab**

Feedback Control

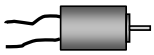


- **Feedback**
  - The Standard Feedback Loop
  - Closed Loop



- **PID**
  - How does each term work?
  - Tuning a PID controller
- **Car example**

- **Motor example**

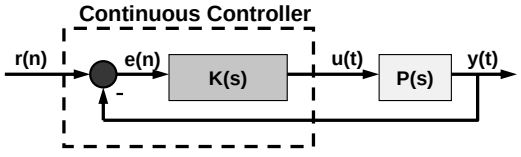


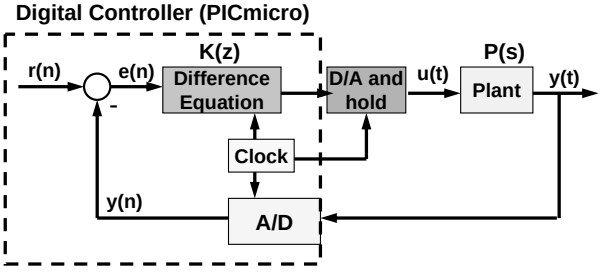
PID Implementation

PID Implementation

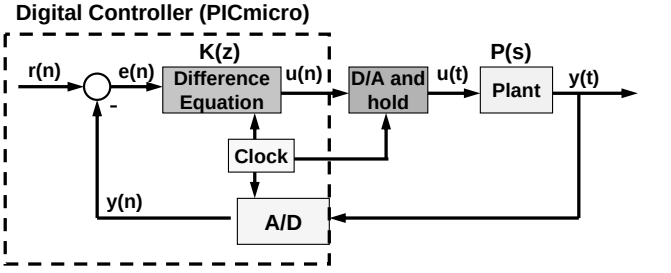
- **The Discrete Feedback loop**
- **The Discrete PID controller**
  - Positional version
  - Velocity
- **PID Controller Realization**
  - Coding flow chart
  - Choosing a sampling and loop update frequency
- **PID LAB**

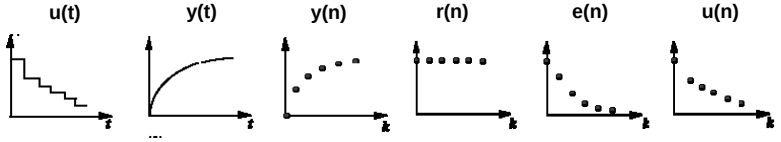
The Discrete Feedback loop





The Discrete Feedback loop





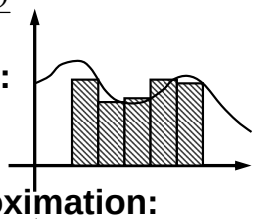
The Discrete PID controller

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

$$u(t) = k_p e(t) + k_i \int_0^t e(t)dt + k_d \frac{de(t)}{dt}$$

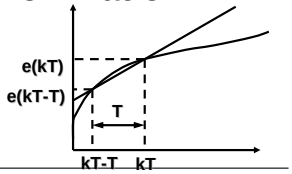
- **Trapezoidal approximation:**

$$\int_0^t e(t)dt \approx \sum_{k=1}^{n} Te(kT)$$



- **Backward difference approximation:**

$$\frac{de(t)}{dt} \approx \frac{e(kT) - e(kT-T)}{T}$$



The Discrete PID controller

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

- **Positional PID controller**

$$u(kT) = K_p e(kT) + K_d \frac{e(kT) - e(kT-T)}{T} + K_i T \sum_{k=1}^{n} e(kT) + u_0$$

The Discrete PID controller

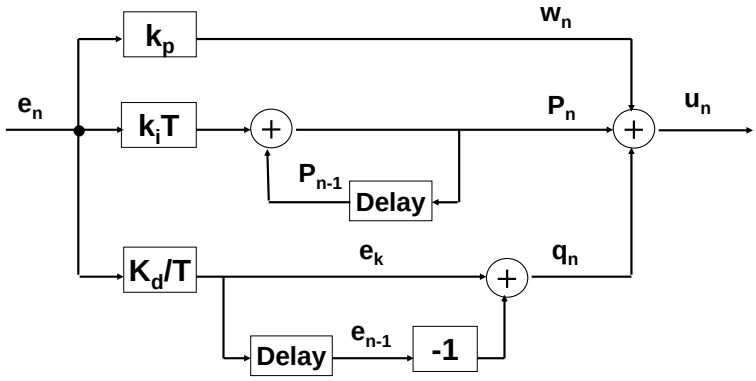
$$\frac{U(s)}{E(s)} = K_p + \frac{K_i}{s} + K_d s$$

- **Velocity PID controller**
  - Shift by one sample and subtract
  - Previous control value is modified
  - Smoother control action with small error

$$u(kT) = u(kT-T) + K_p[e(kT) - e(kT-T)] + K_i T e(kT)$$
$$+ \frac{Kd}{T}\left[e(kT) - 2e(kT-T) - e(kT-2T)\right]$$

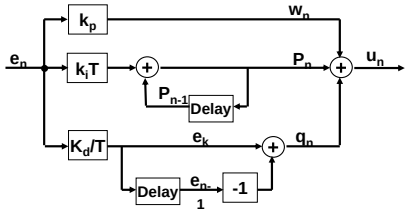
PID Controller Realization

- PID controller as a parallel structure:



PID Controller Realization

- PID controller as a parallel structure:



PID Update:

$$w_n = k_p e_n$$

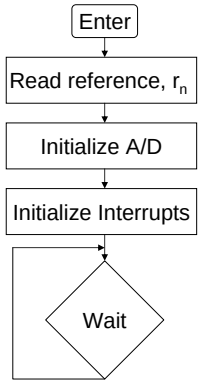
$$p_n = k_i T e_n + p_{n-1}$$

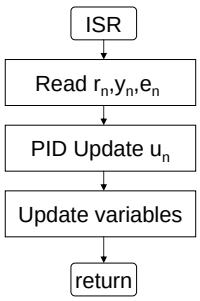
$$q_n = \frac{k_d}{T}(e_n - e_{n-1})$$

$$u_n = w_n + p_n + q_n$$

Code Blocks

- **Typical code flow chart:**





Choosing a sampling rate

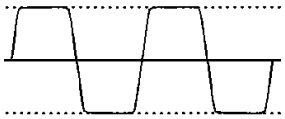
- If plant dominant time constant is T_p:
  - $T < T_p/10$
- If Ziegler-Nichols open-loop model was used:

$$P(s) = \frac{Ke^{-sT_d}}{1+sT_1}$$

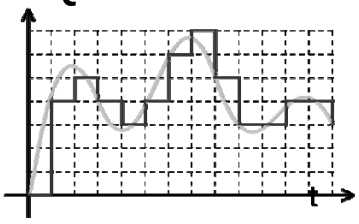
  - $T < T_1/4$

Things to Watch Out For

- **Saturation and Integral Wind Up**



- **Noise and Quantization**

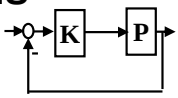


LAB

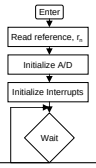
Summary

- **Introduction to Control Systems**
- **Modeling**
  - Examples

- **Feedback Control Systems**
  - Examples



- **PID Implementation**
  - LAB



References

- "Microcontroller Based Applied Digital Control"
  - by Dogan Ibrahim
- "Schaum's Outline of Feedback and Control Systems"
  - by Allen Stubberud
- "Applied Control Theory for Embedded Systems"
  - by Tim Wescott
- "Modern Control Design With MATLAB and SIMULINK"
  - by Ashish Tewari

References

- "Discrete-Time Control Systems"
  - by Katsuhiko Ogata
- "Control Tutorials for MATLAB and Simulink: A Web-Based Approach"
  - by William Messner
- AN964 "software PID Control of an Inverted Pendulum Using the PIC16F684"
- AN937 "Implementing a PID Controller Using a PIC18 MCU"
- "Control system" wikipedia.org