CprE 488 – Embedded Systems Design

Lecture 7 – Embedded Control Systems

Joseph Zambreno and Phillip Jones
Electrical and Computer Engineering
Iowa State University

www.ece.iastate.edu/~zambreno www.ece.iastate.edu/~phjones
rcl.ece.iastate.edu

If everything seems under control, you're just not going fast enough. – Mario Andretti

Motivation

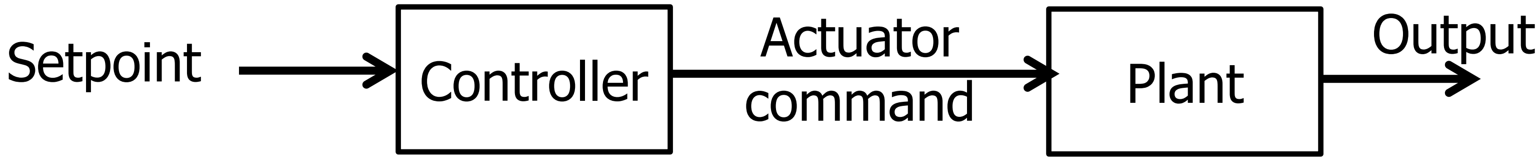
- Embedded systems often have a control aspect
  - Cruise control: Maintain a speed
  - Quadcopter control: Maintain a hover
  - Heating system: Maintain a given temperature
- How to direct a system to reach a given goal (i.e. setpoint)?
- What are some important properties of a feedback-driven control system?

Terminology

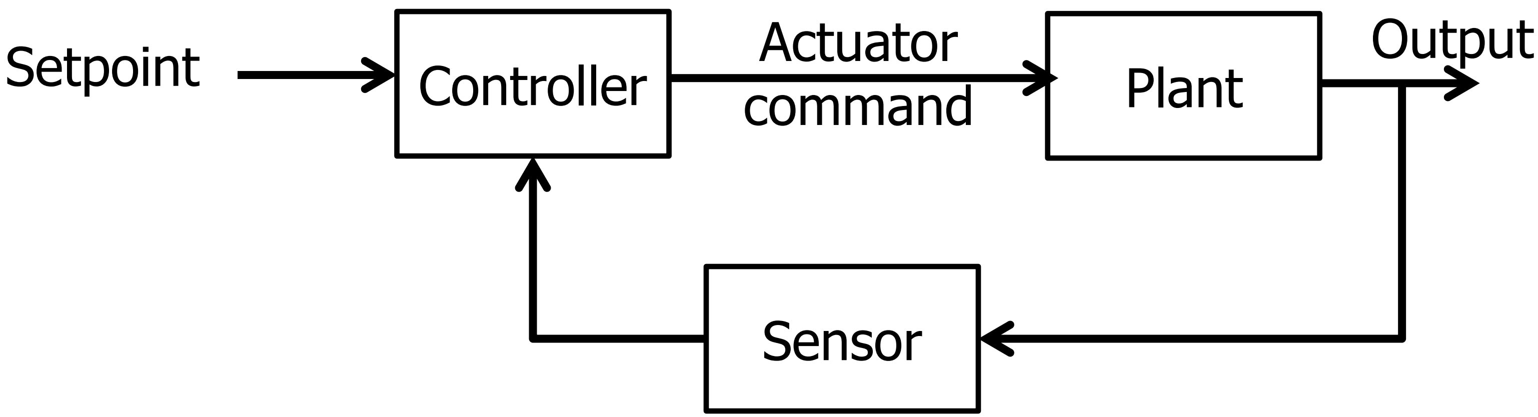
- Plant/Process: system being controlled
  - car, plane, building,
- Sensor: mechanism for measuring quantities of the system
  - thermometer, barometer, tachometer, encoder, accelerometer
- Actuator: mechanism to enact change on the plant
  - servo, valve, muscle
- Setpoint: goal value of the quantity being controlled
  - Speed, temperature, height
- Controller: mechanism to process senor signals and command actuators
  - Microprocessor
- Control Law: Rule for mapping sensor signals to actuator commands
  - On-off, P, PD, PI, PID, State-space, …

Terminology

- Open-Loop Control Systems utilize a controller or control actuator to obtain the desired response.

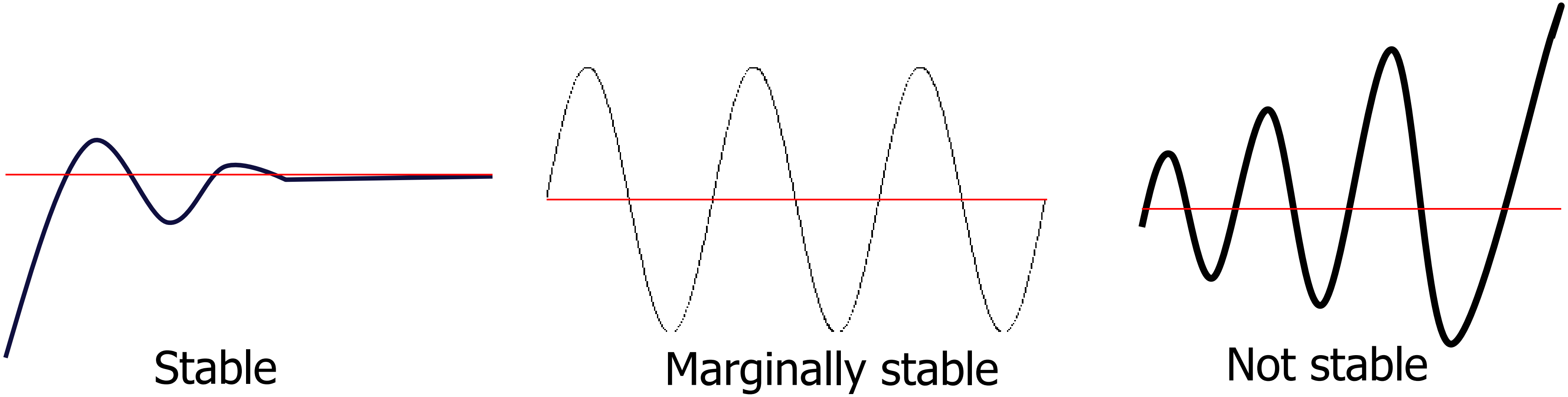


- Closed-Loop Control Systems utilizes feedback to compare the actual output to the desired output response.



Typical Controller Metrics

- Stability (e.g. bounded oscillation of system output)



- For a stable controlled system
  - Disturbance Rejection: How well does system hold setpoint in the presence of a disturbance (e.g. shoving the quad on the turn table)
  - Command tracking: How well does the system respond to changes in the controller setpoint
    - Rise time
    - Settling time

Examples

- Watt's fly ball governor (1788)

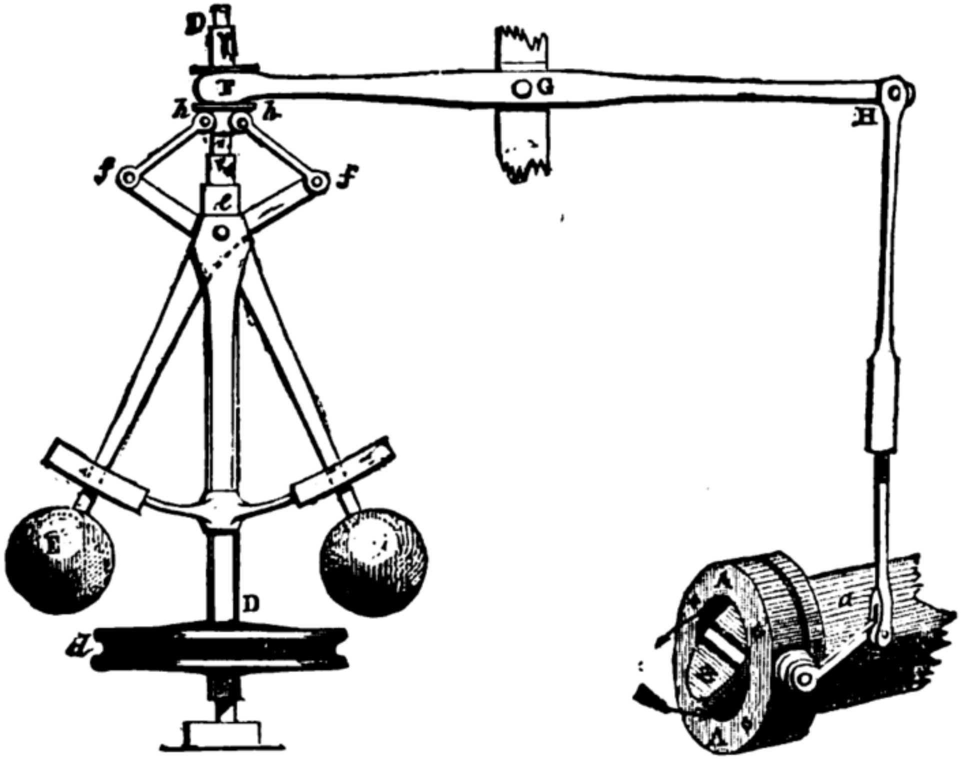
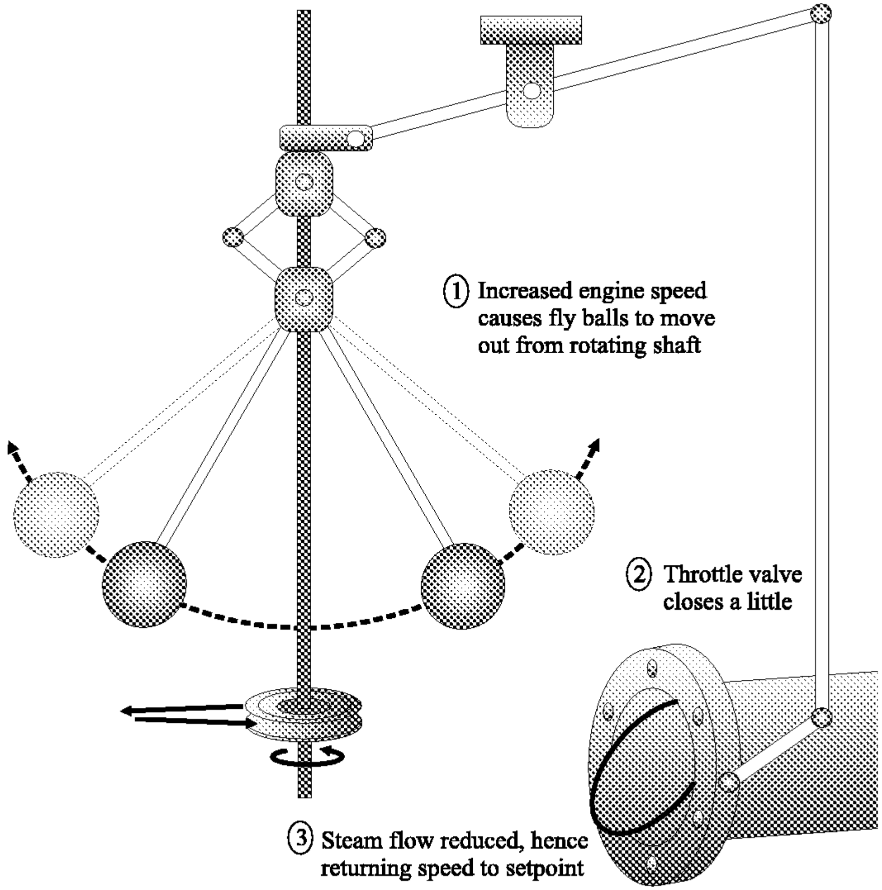


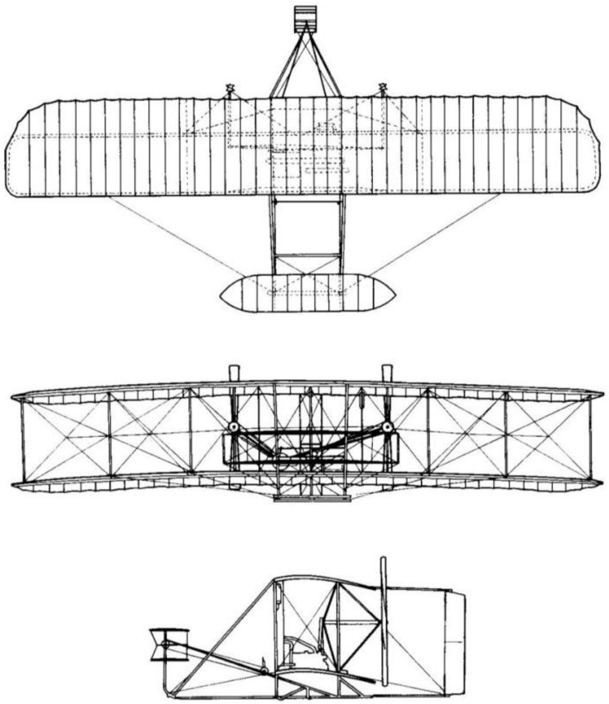
FIG. 4.—*Governor and Throttle-Valve.*



- 1868: James Clerk Maxwell publishes the first theoretical study of steam engine governors. By that time, there were more than 75,000 governors installed in England.

Examples (cont.)

- Orville and Wilbur Wright made the first successful experiment with manned flight (1905)



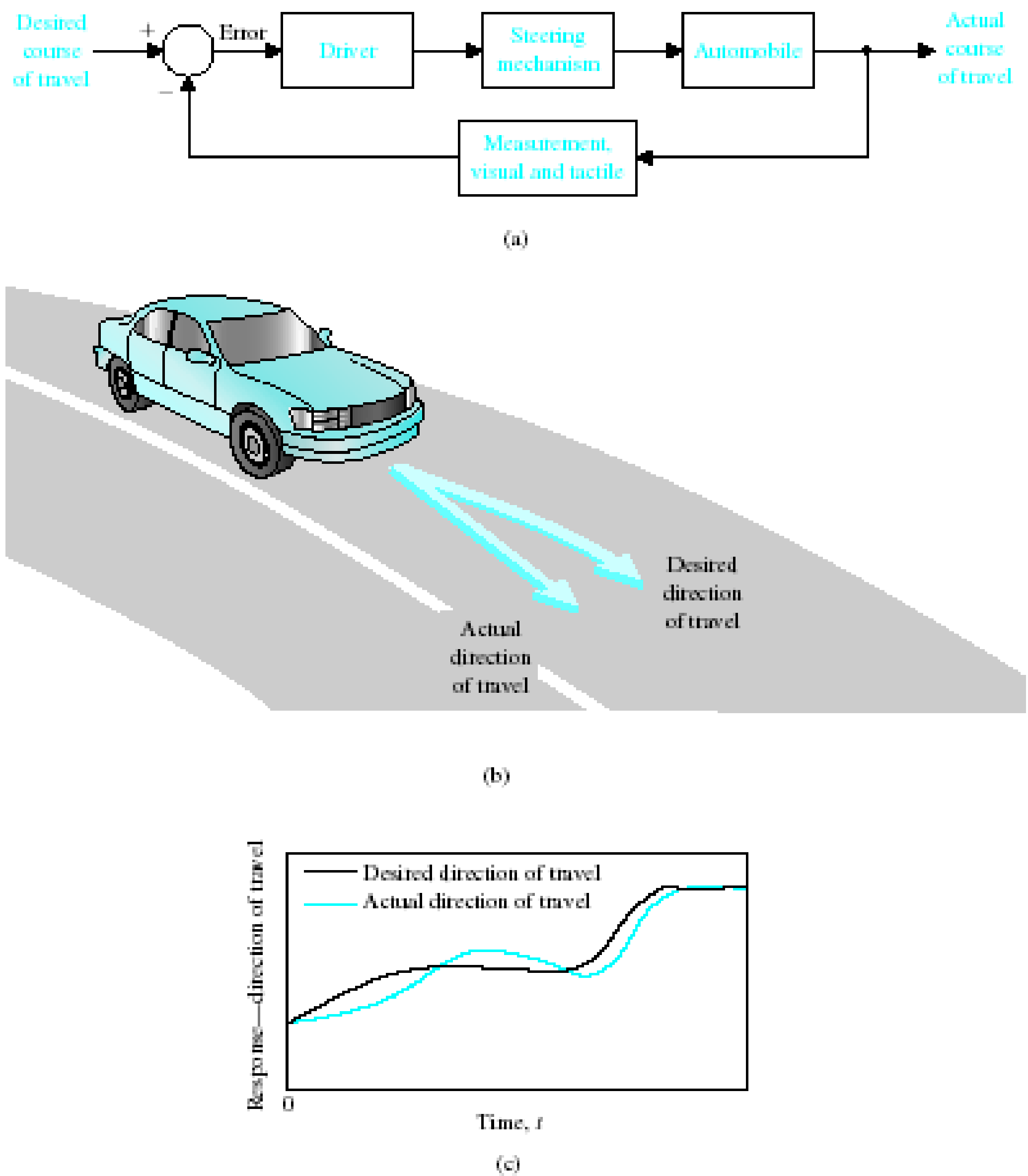
- Their main insight was that the airplane itself had to be inherently unstable, which would give the pilot more control and render the overall flying system (pilot and machine) stable
- The first autopilot was developed by Sperry Corp. in 1912

Examples

- Automobile steering control system.

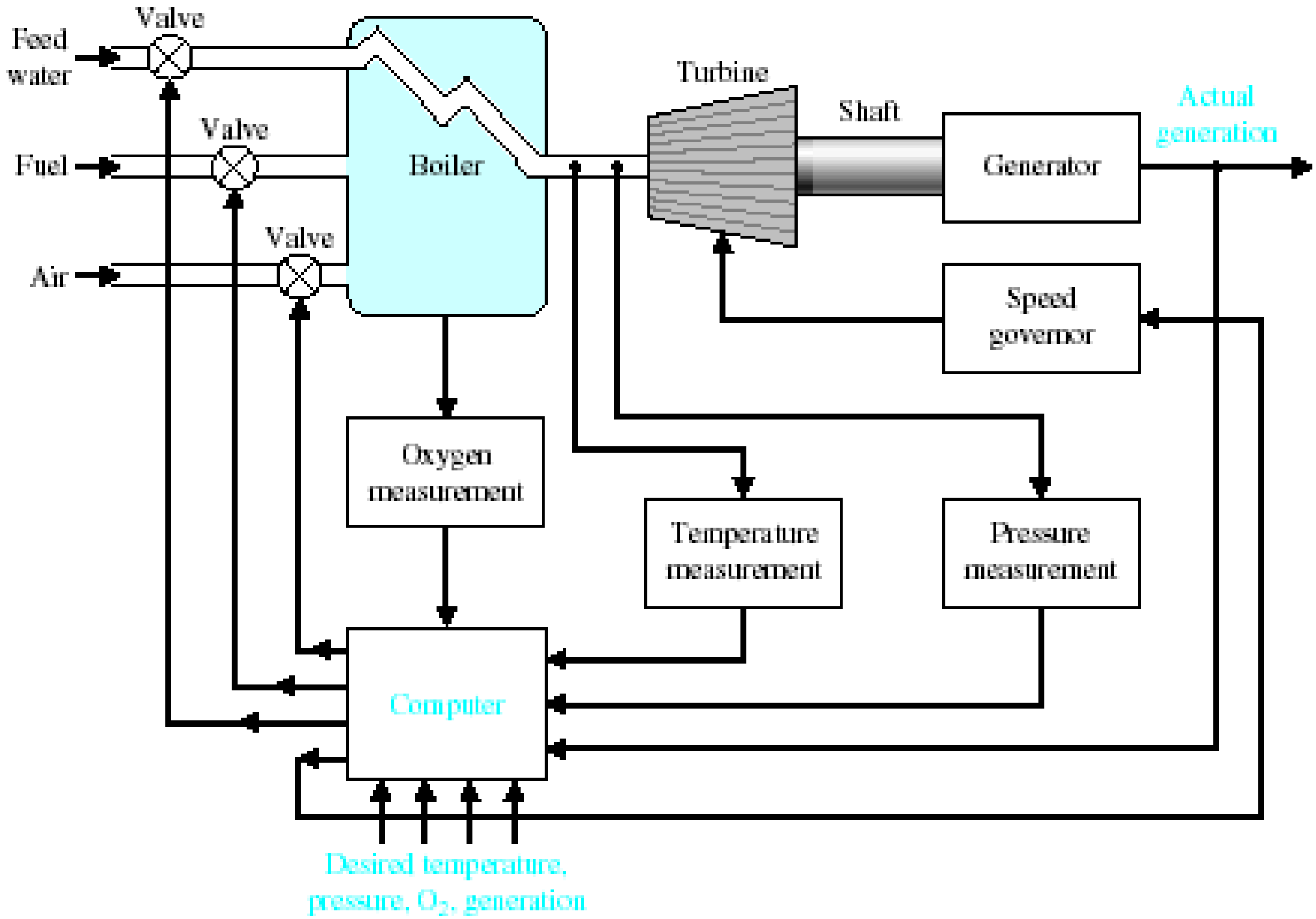
- The driver uses the difference between the actual and the desired direction of travel to generate a controlled adjustment of the steering wheel.

- Typical direction-of-travel response.



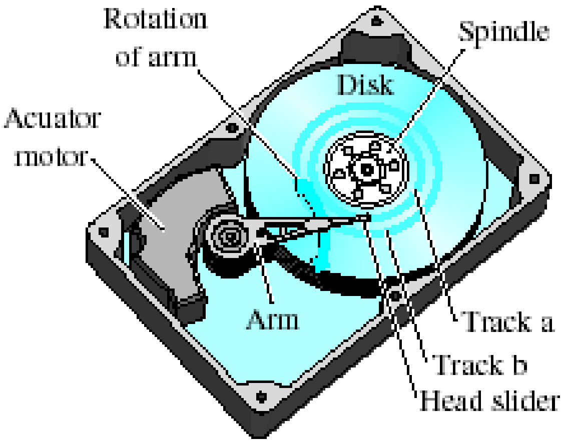
Examples (cont.)

- Boiler Generator



Examples (cont.)

- Hard drive head control

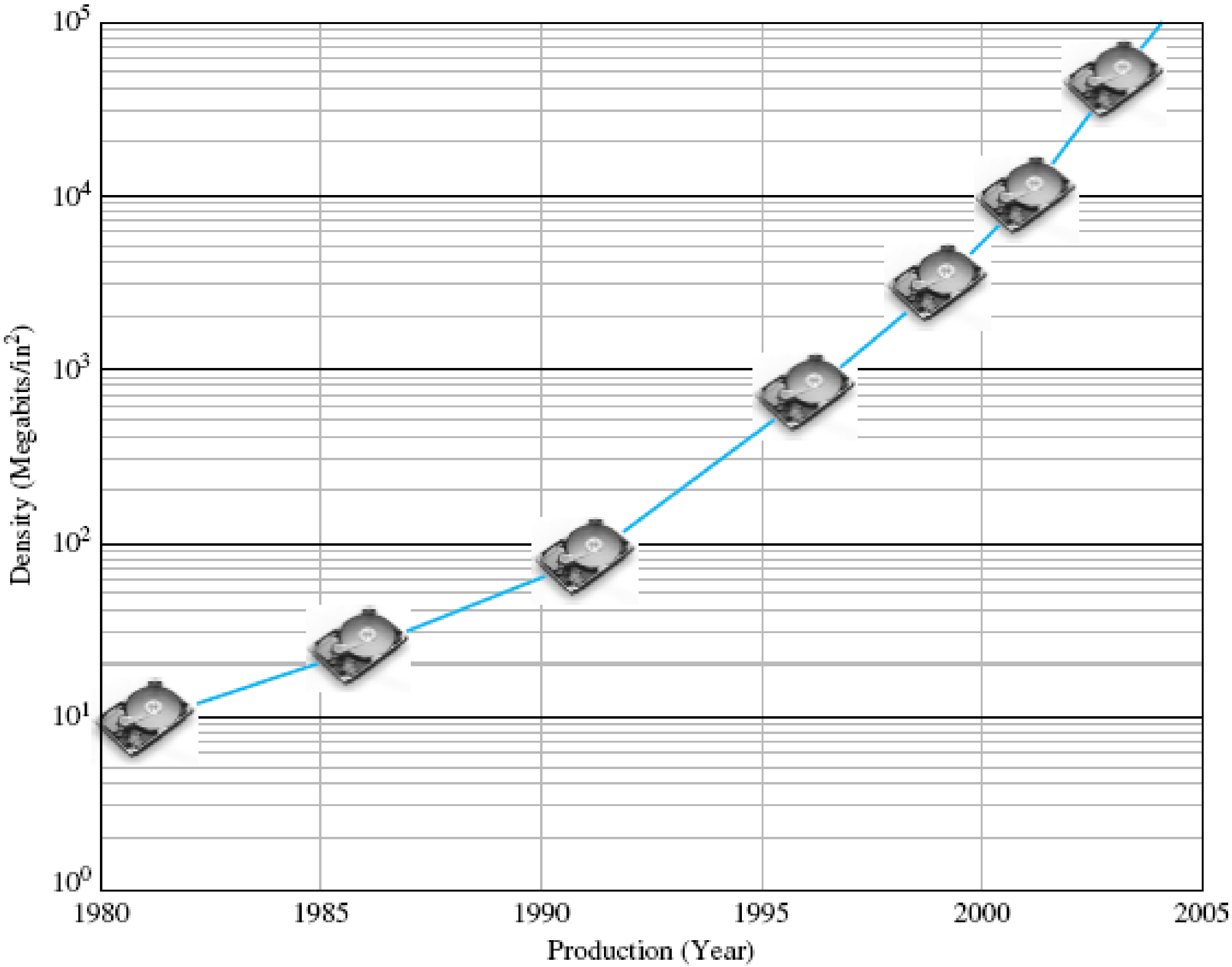


(a) (b)

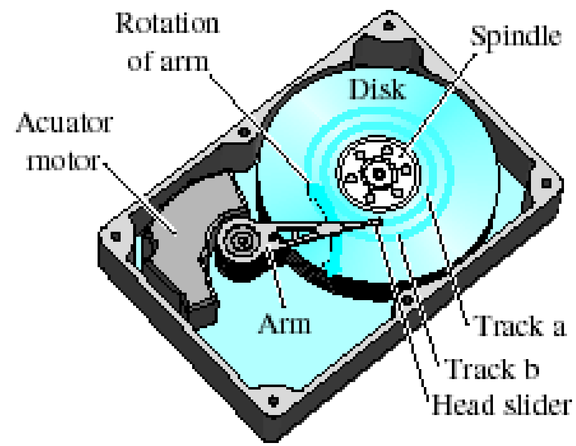
(a) A disk drive ©1999 Quantum Corporation. All rights reserved.
(b) Diagram of a disk drive.

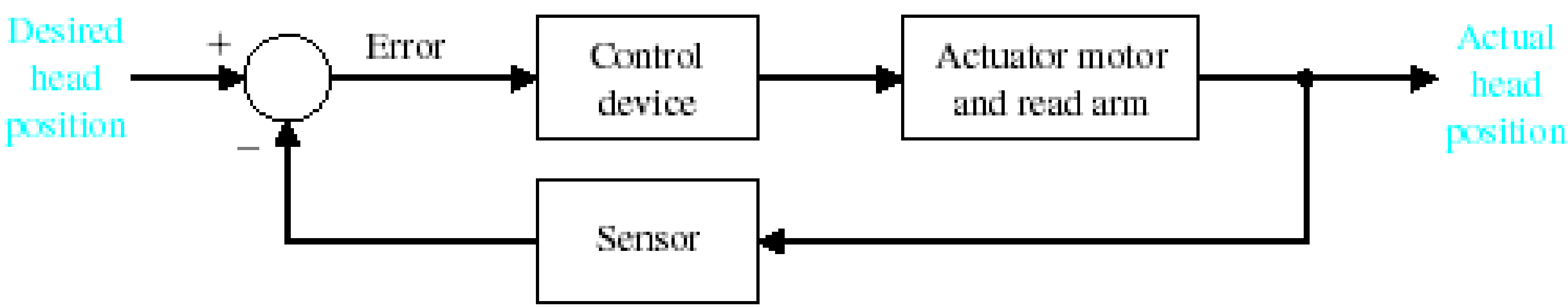
Examples (cont.)

- Hard drive head control



- Hard drive head control





PID control

- Continuous-time and Discrete-time form

$$u(t) = K_P e(t) + K_I \int_0^t e(t)\,\mathrm{d}t + K_D \frac{\mathrm{d}e(t)}{\mathrm{d}t}$$

$$u[n] = K_P e[n] + K_I \sum_{j=0}^{n} e[j] + K_D(e[n] - e[n-1])$$

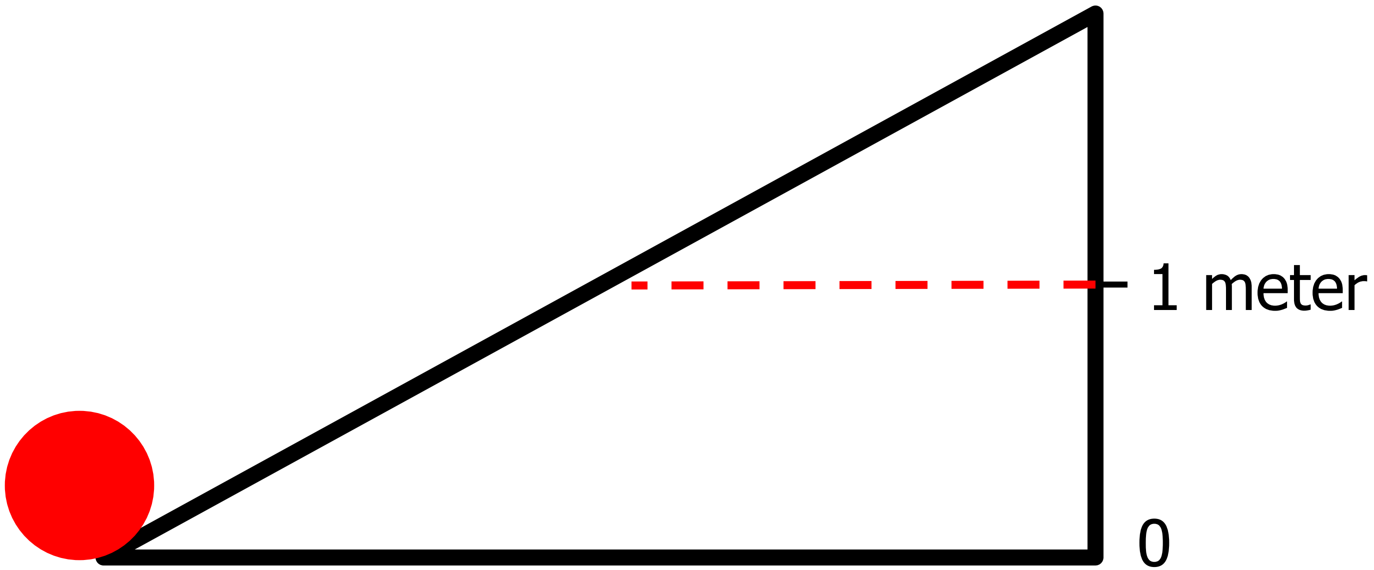
—$u(t), u[n]$ is the correction given by the controller to the system at time t or discrete sample n;

—$e(t), e[n]$ is the error between the set point and current state of the system under control at time t or discrete sample n;

—K_P, K_I, and K_D scale the error, integral (sum) of error, and derivative (difference) of the error, respectively.

PID Plot Analysis

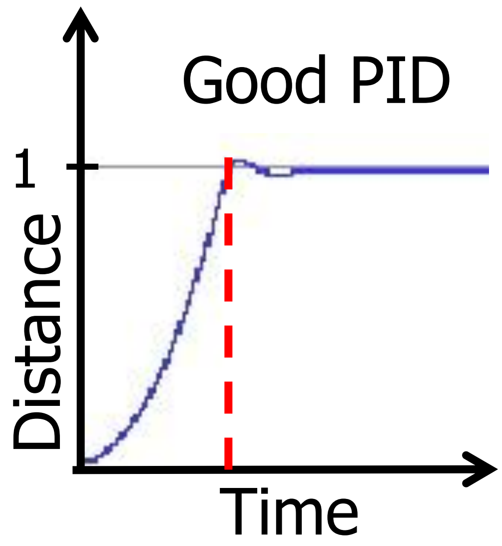
- Practice intuition for PID tuning

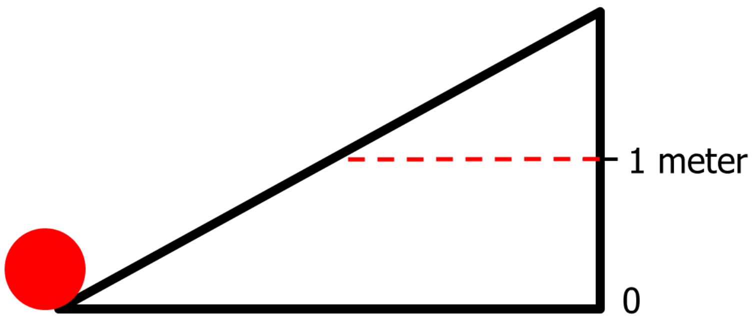


PID Plot Analysis (cont.)

https://sites.google.com/site/fpgaandco/pid

- Car max angle
- P = 30, I=1, D=2.2
- M = .2 Kg, Damping force = 0, Motor force limit 1 N

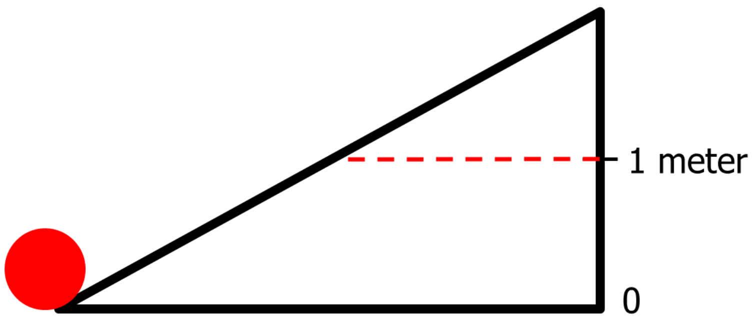


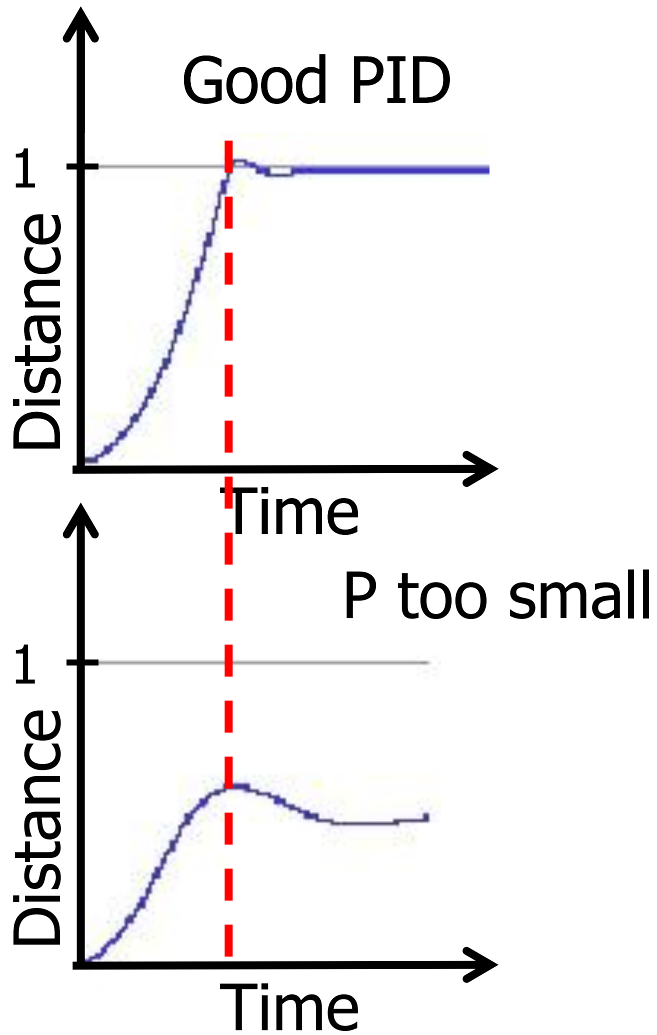


PID Plot Analysis (cont.)

https://sites.google.com/site/fpgaandco/pid

- Car max angle
- P = 30, I=1, D=2.2
- M = .2 Kg, Damping force = 0, Motor force limit 1 N

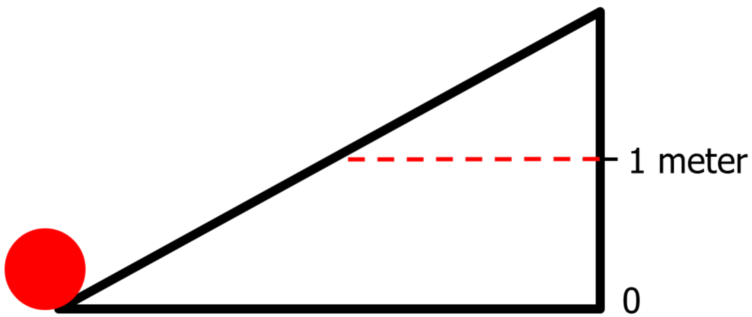


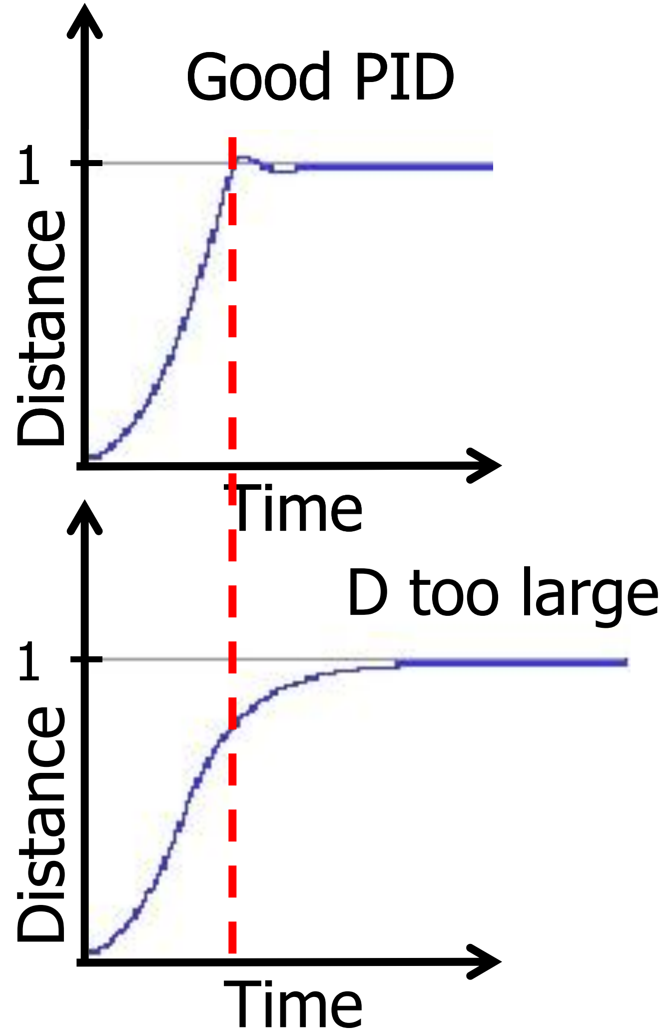


PID Plot Analysis (cont.)

https://sites.google.com/site/fpgaandco/pid

- Car max angle
- P = 30, I=1, D=2.2
- M = .2 Kg, Damping force = 0, Motor force limit 1 N

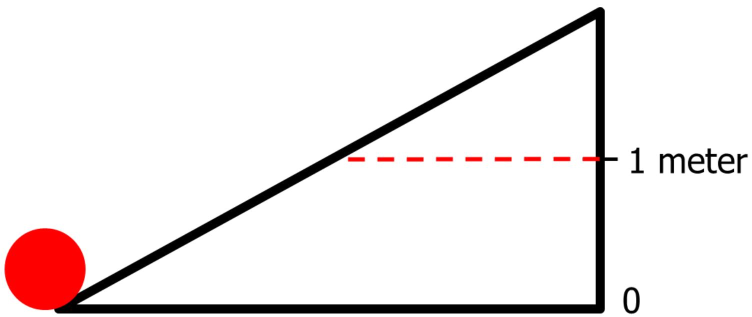


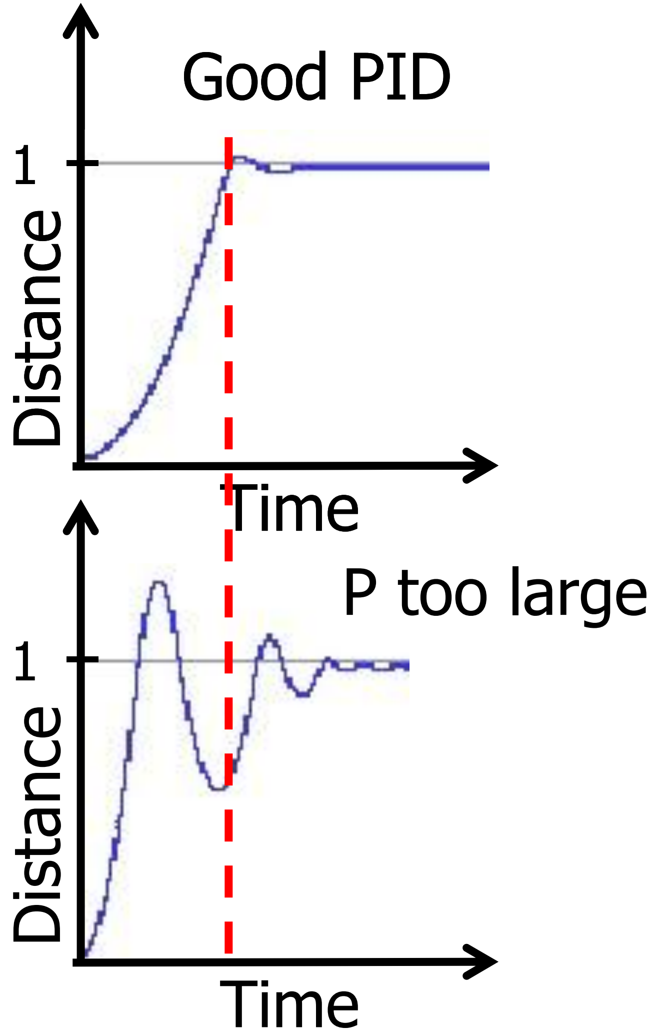


PID Plot Analysis (cont.)

https://sites.google.com/site/fpgaandco/pid

- Car max angle
- P = 30, I=1, D=2.2
- M = .2 Kg, Damping force = 0, Motor force limit 1 N

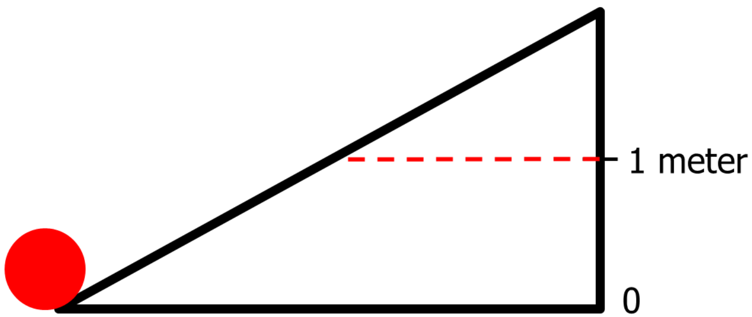


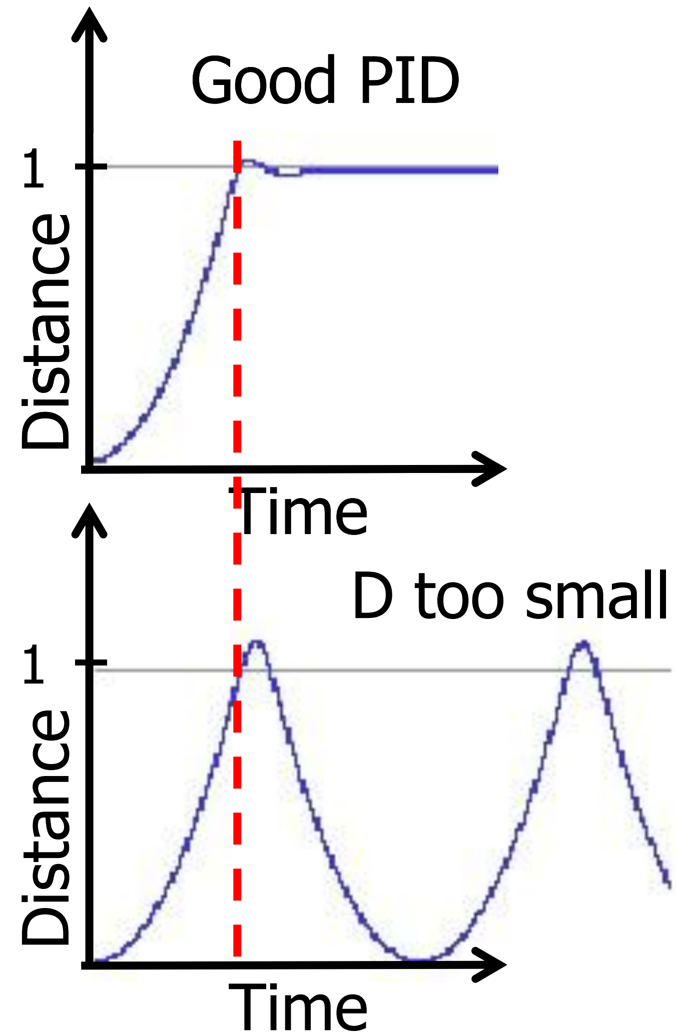


PID Plot Analysis (cont.)

https://sites.google.com/site/fpgaandco/pid

- Car max angle
- P = 30, I=1, D=2.2
- M = .2 Kg, Damping force = 0, Motor force limit 1 N

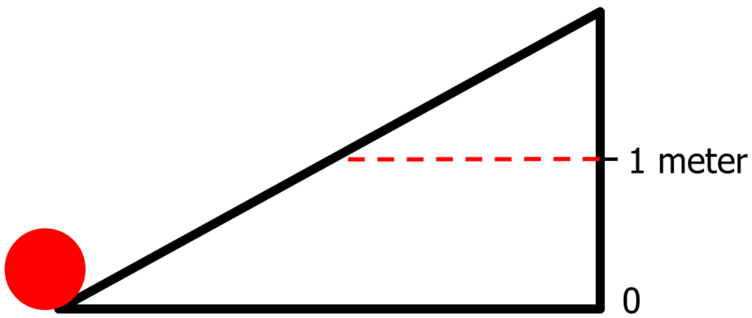


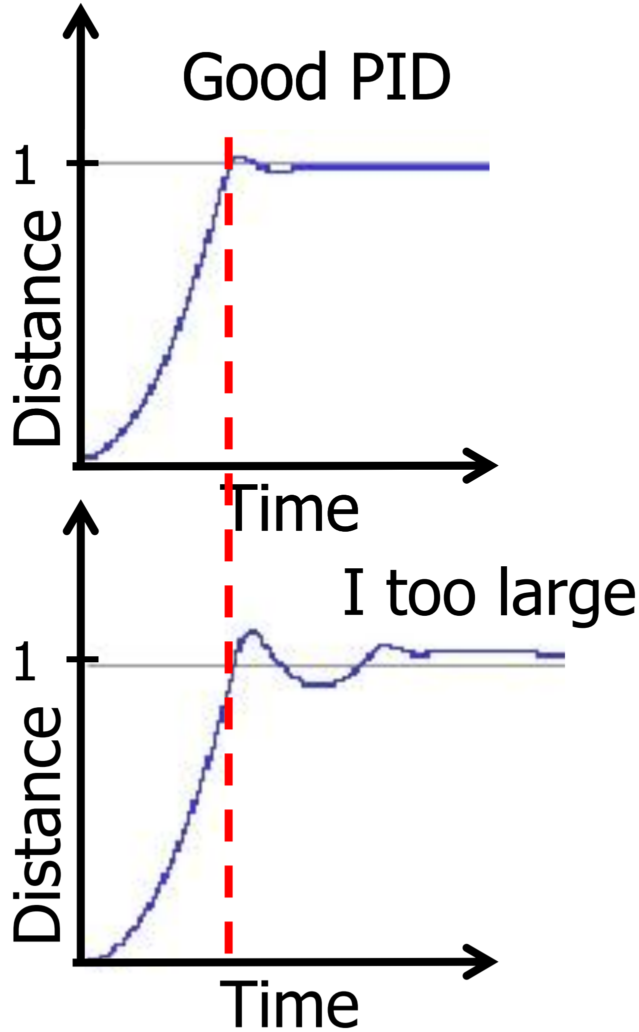


PID Plot Analysis (cont.)

https://sites.google.com/site/fpgaandco/pid

- Car max angle
- P = 30, I=1, D=2.2
- M = .2 Kg, Damping force = 0, Motor force limit 1 N

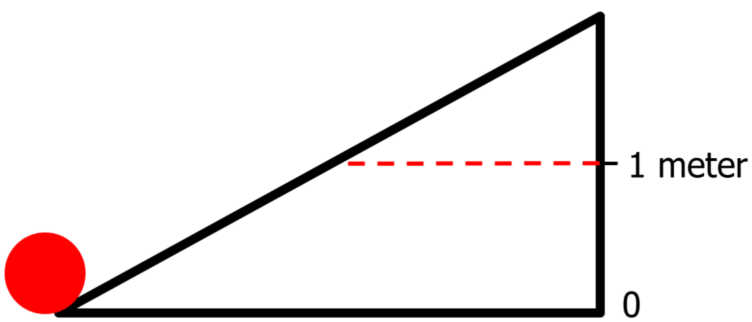




PID Plot Analysis (cont.)

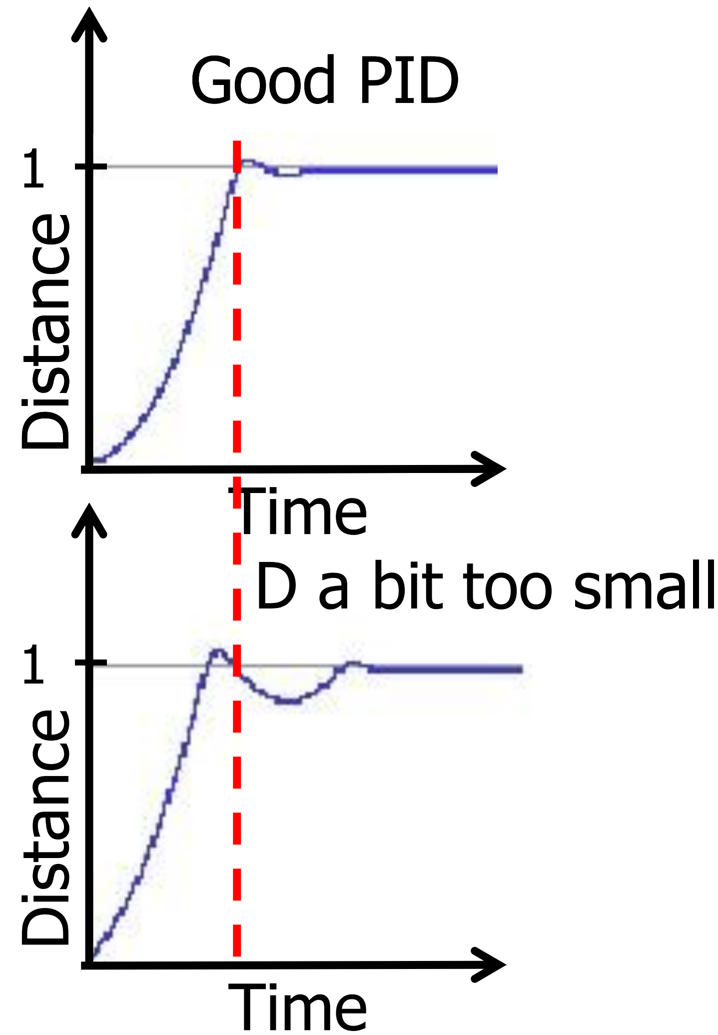
https://sites.google.com/site/fpgaandco/pid

- Car max angle
- P = 30, I=1, D=2.2
- M = .2 Kg, Damping force = 0, Motor force limit 1 N

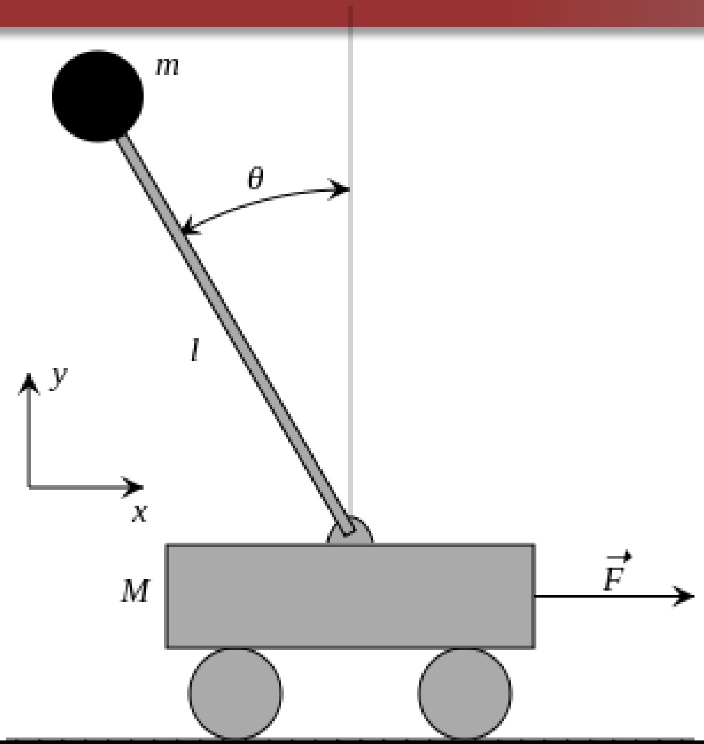


https://sites.google.com/site/fpgaandco/pid

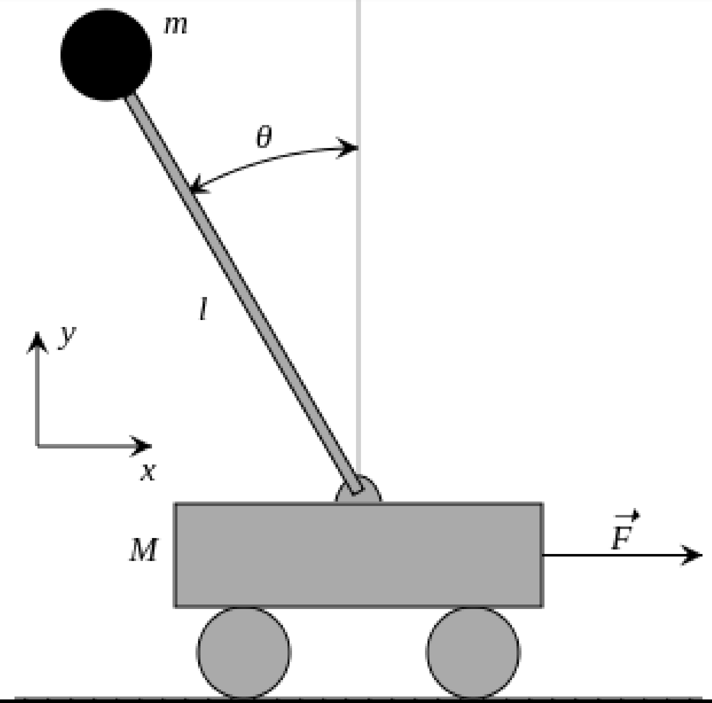
- Car max angle
- P = 30, I=1, D=1.5
- M = .2 Kg, Damping force = 0, Motor force limit 1 N

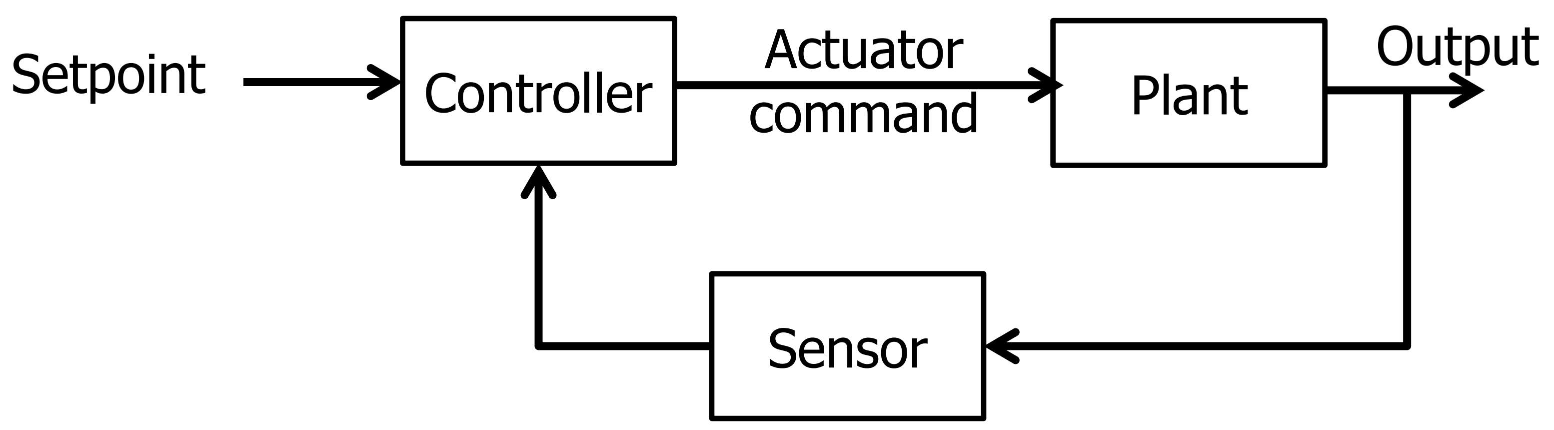


Inverted Pendulum

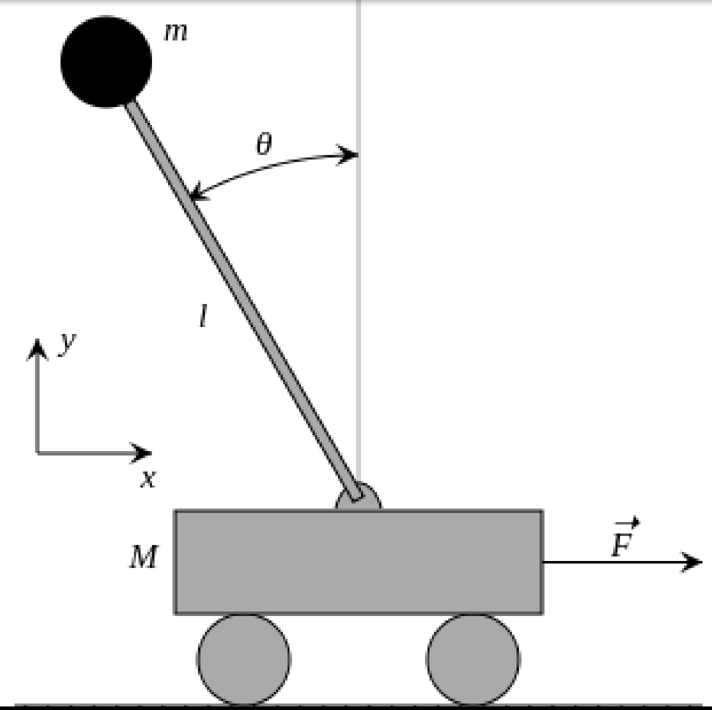


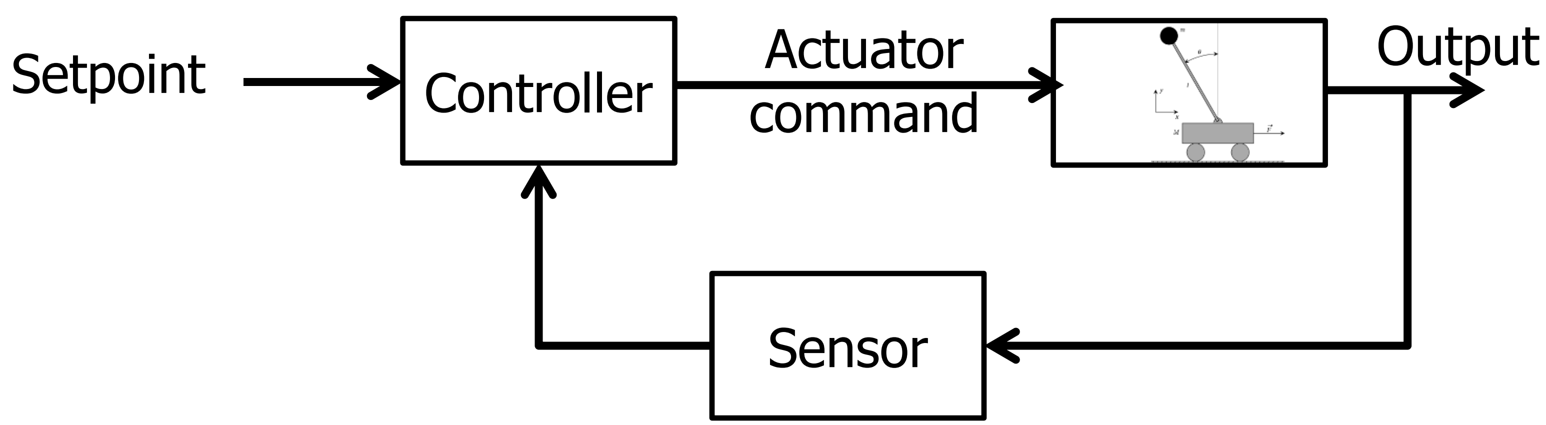
Inverted Pendulum (cont.)



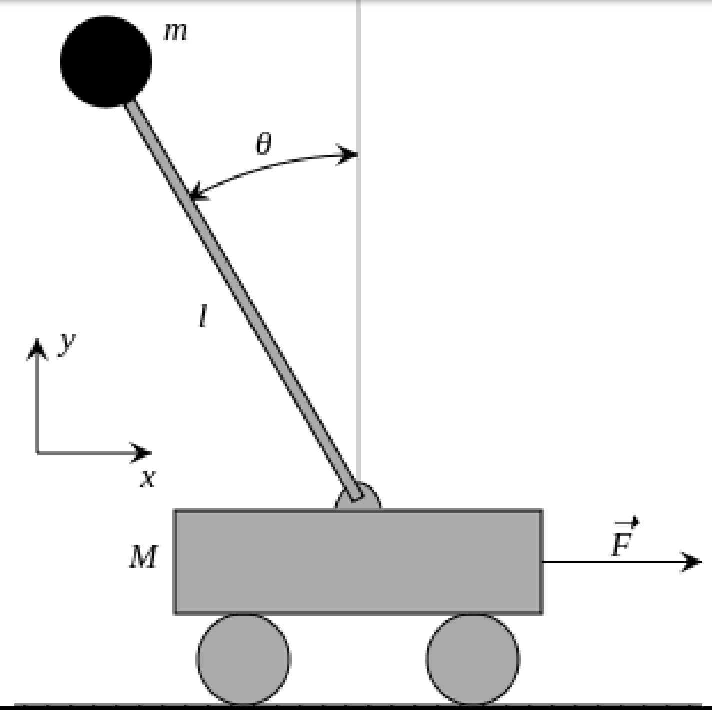


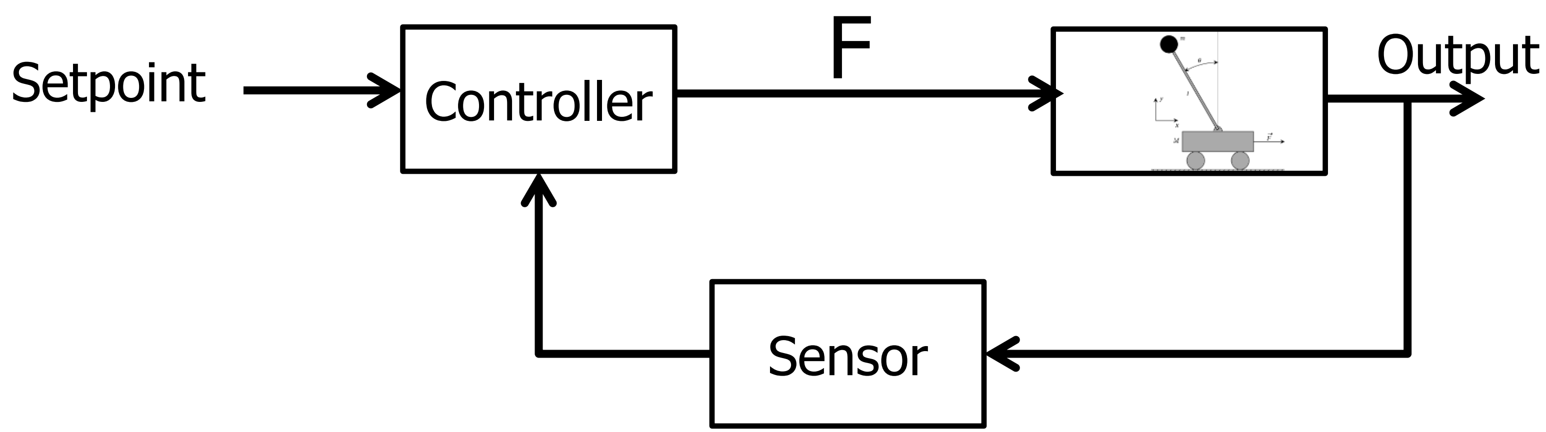
Inverted Pendulum (cont.)



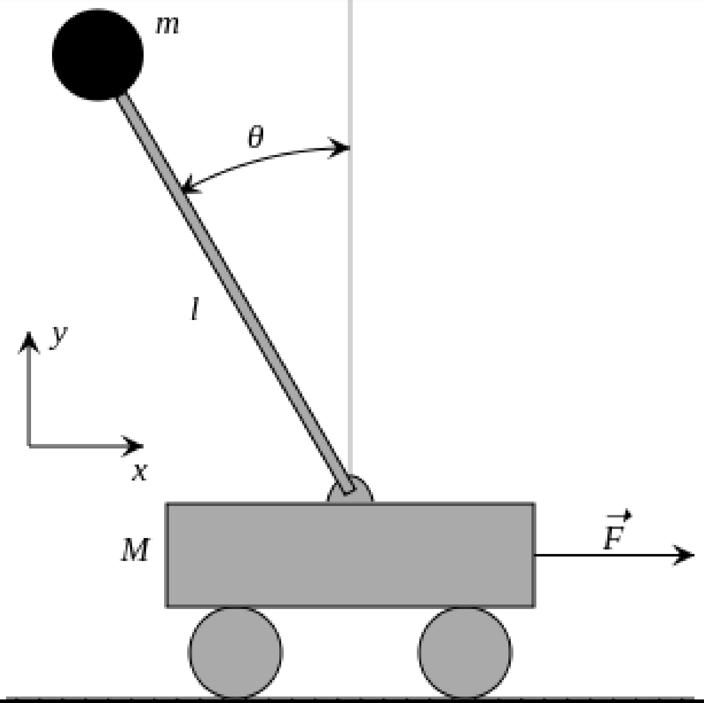


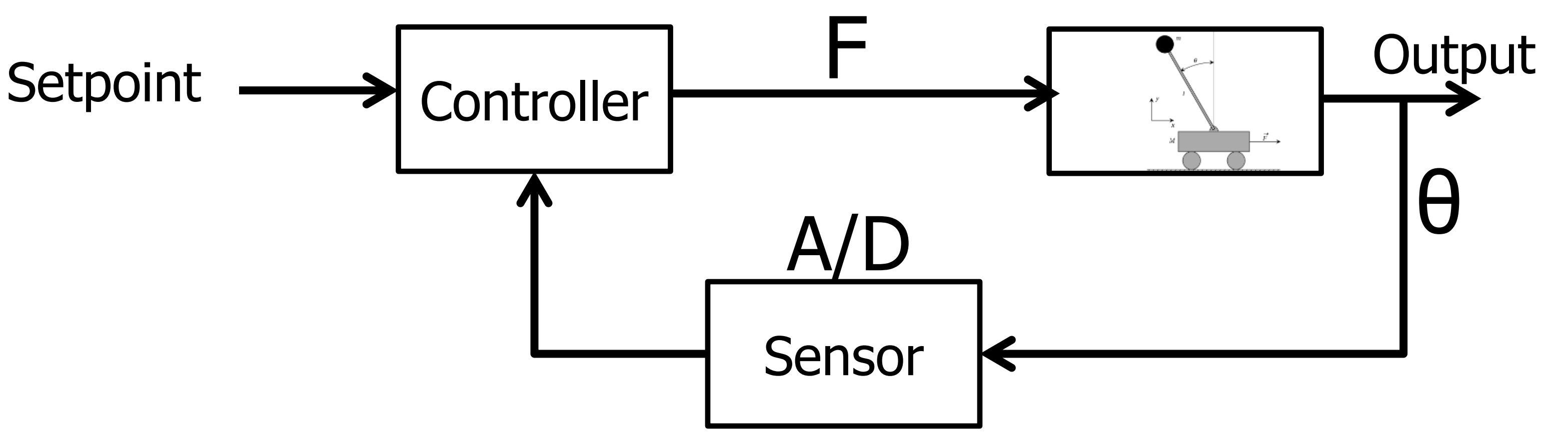
Inverted Pendulum (cont.)



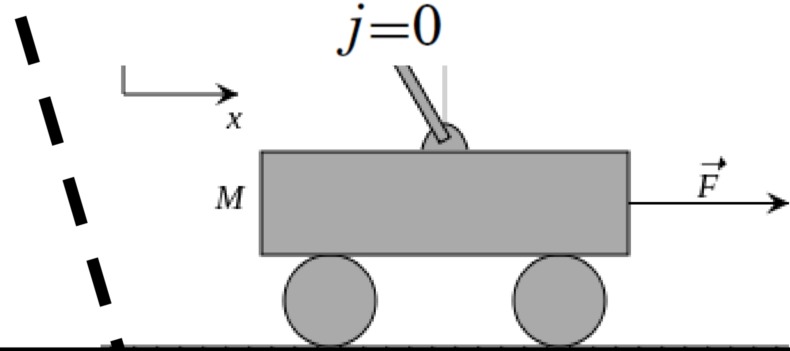


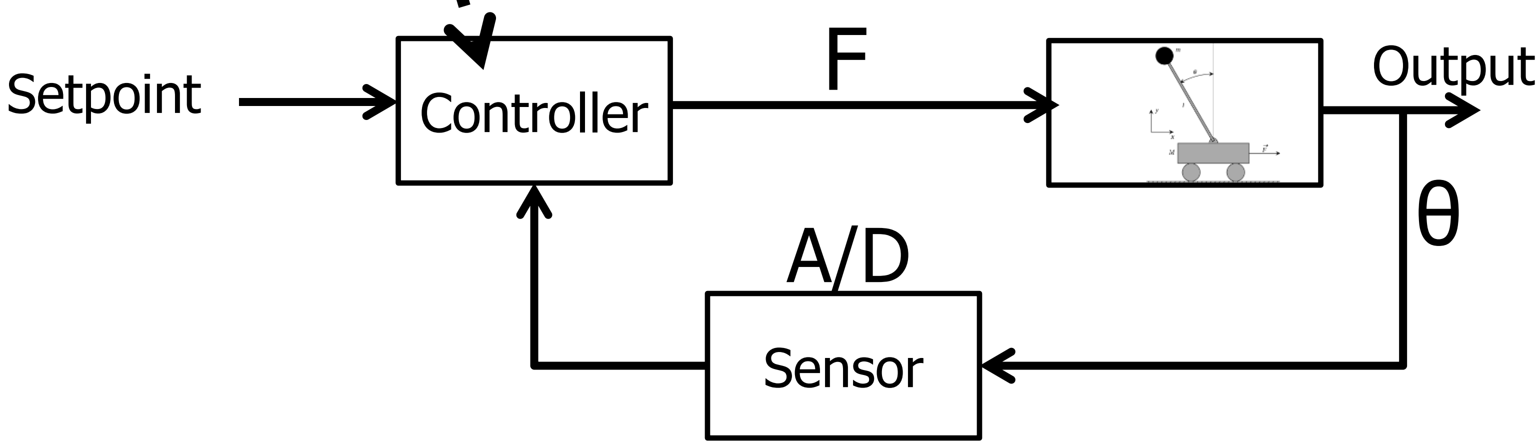
Inverted Pendulum (cont.)



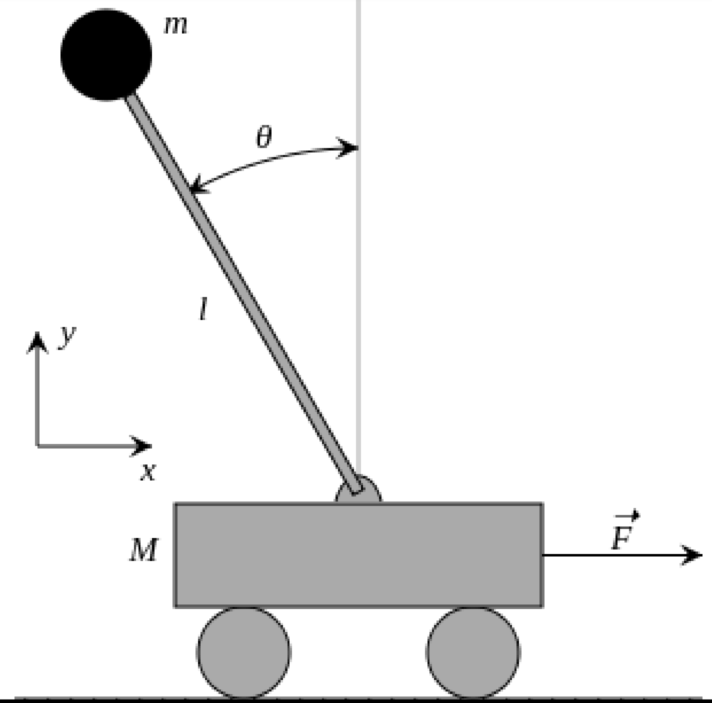


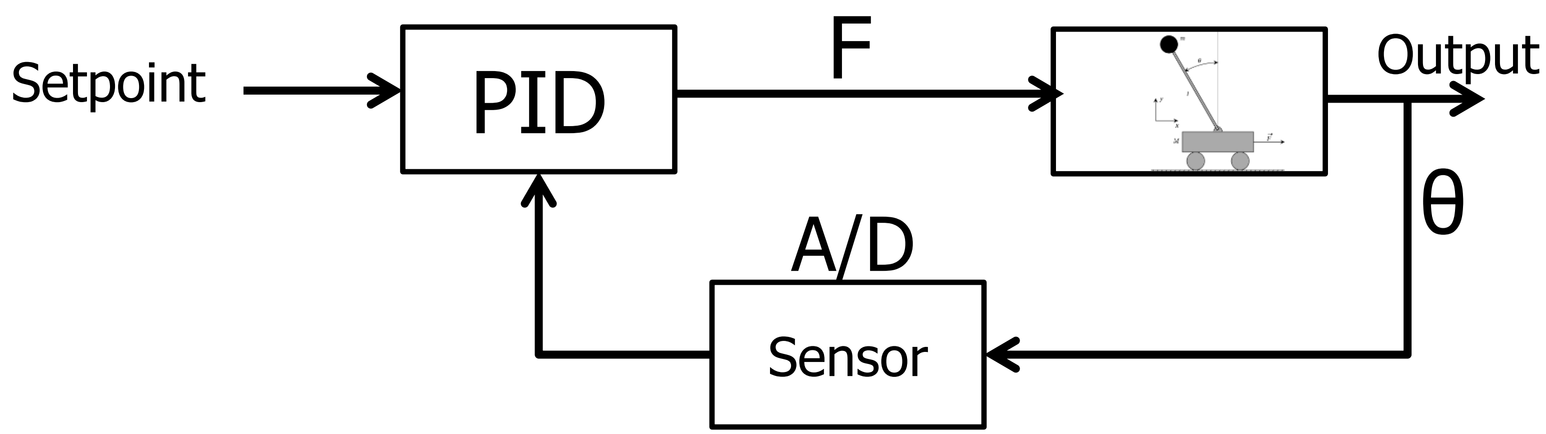
$$u[n] = K_P e[n] + K_I \sum_{j=0}^{n} e[j] + K_D(e[n] - e[n-1])$$



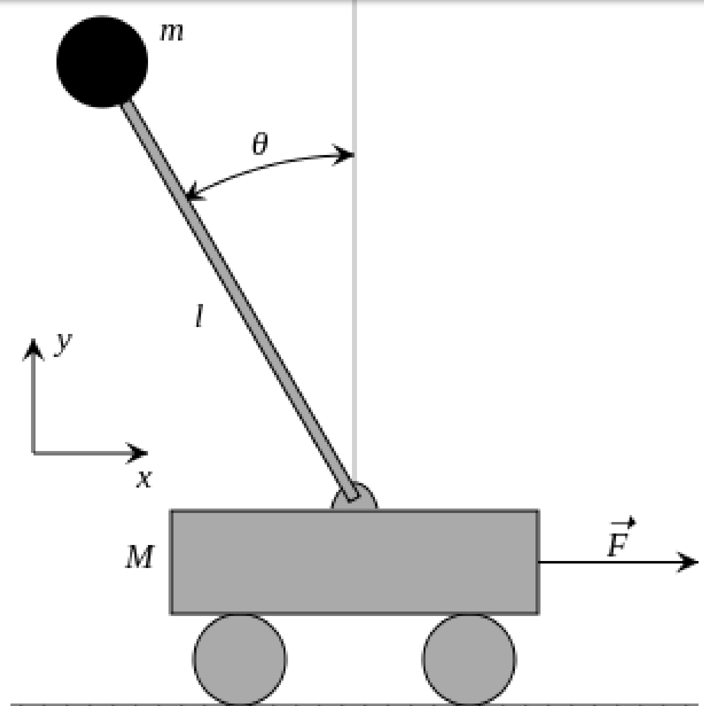


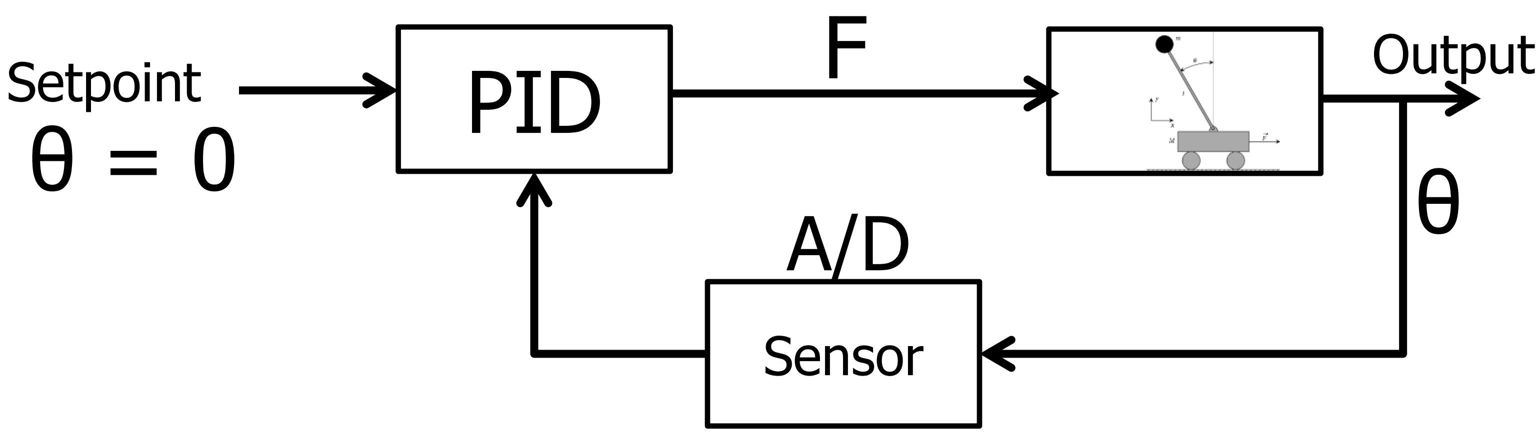
Inverted Pendulum (cont.)



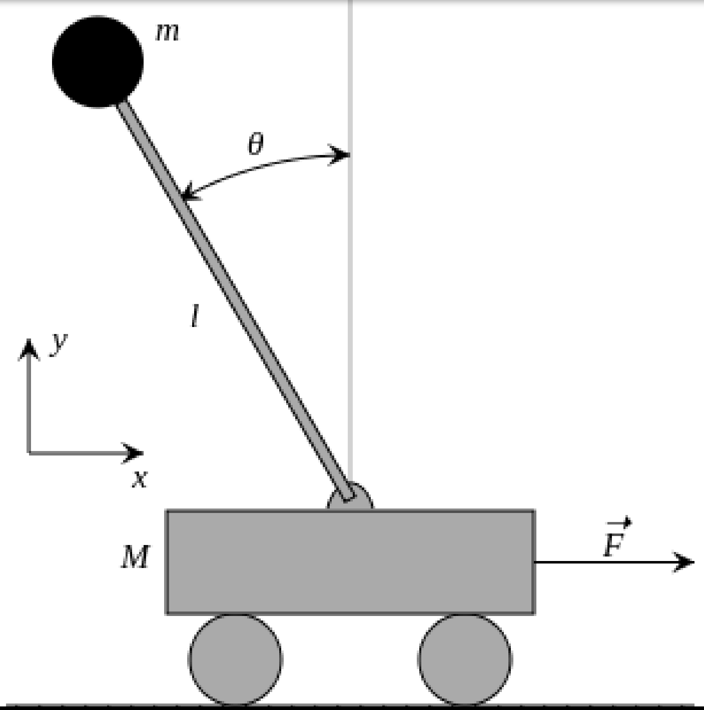


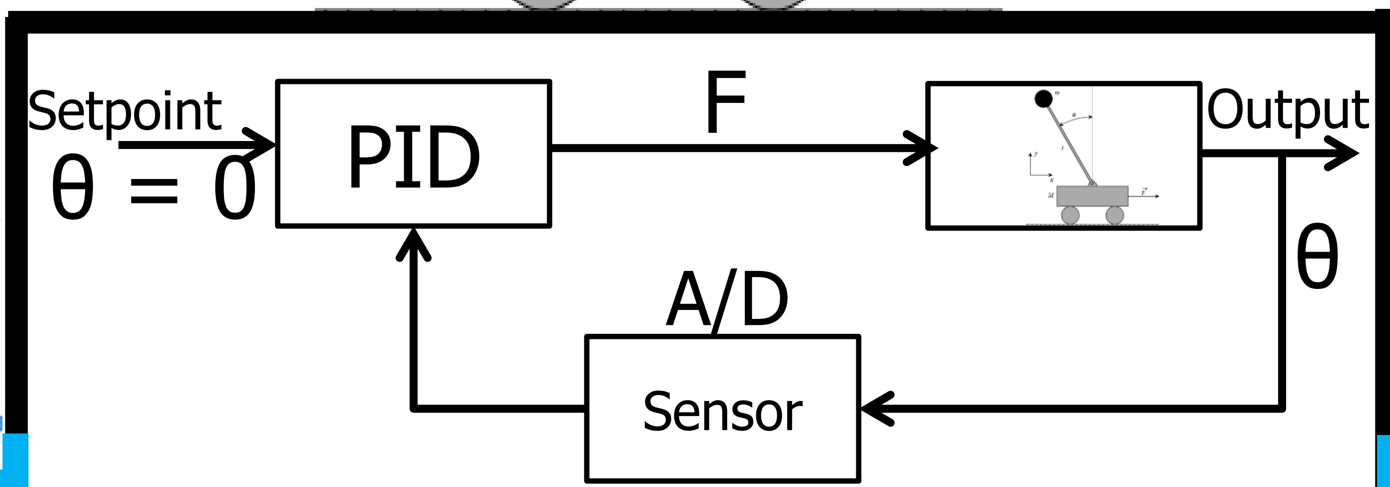
Inverted Pendulum (cont.)



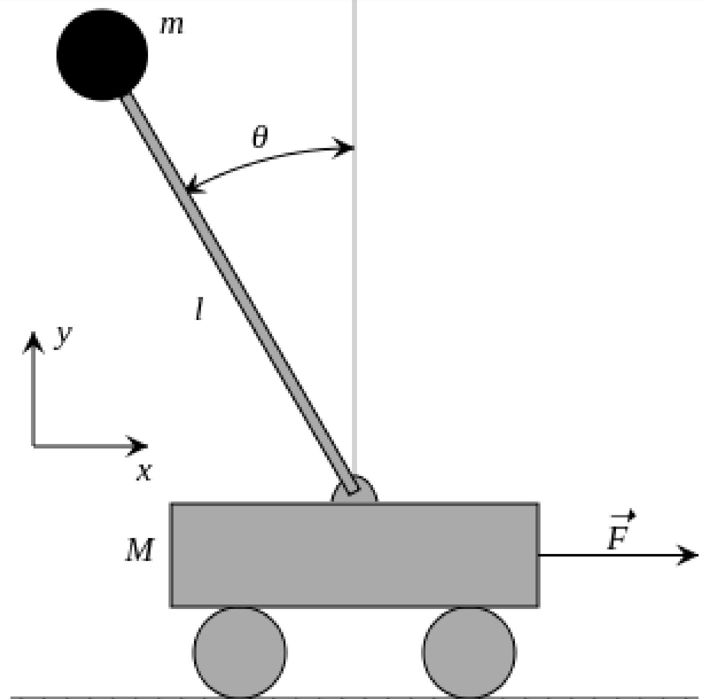


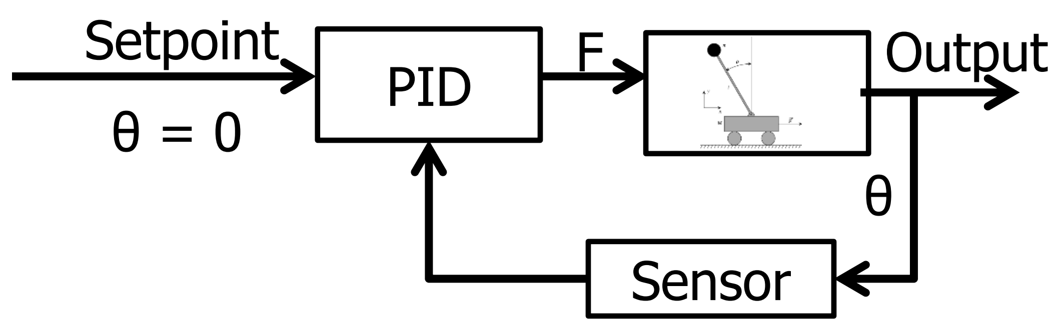
Inverted Pendulum (cont.)



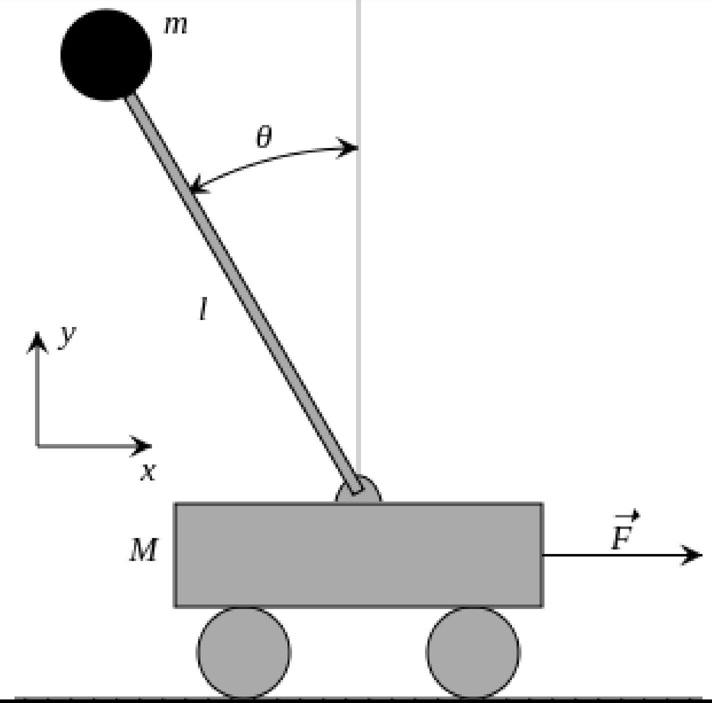


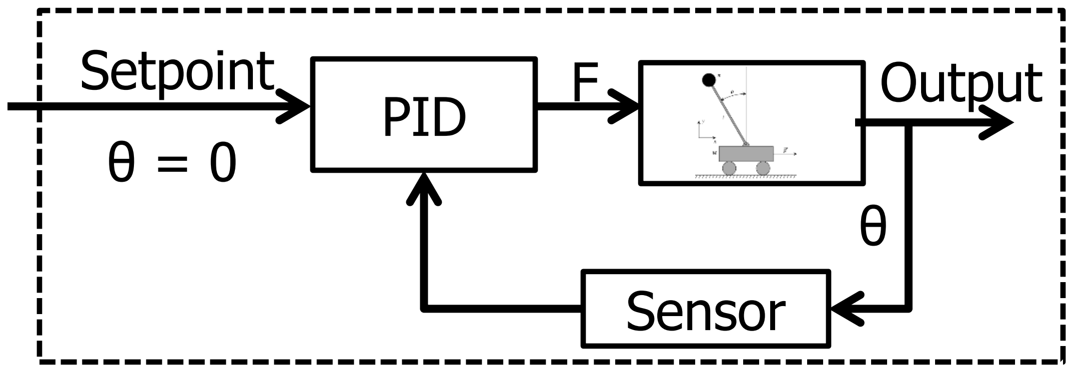
Inverted Pendulum (cont.)



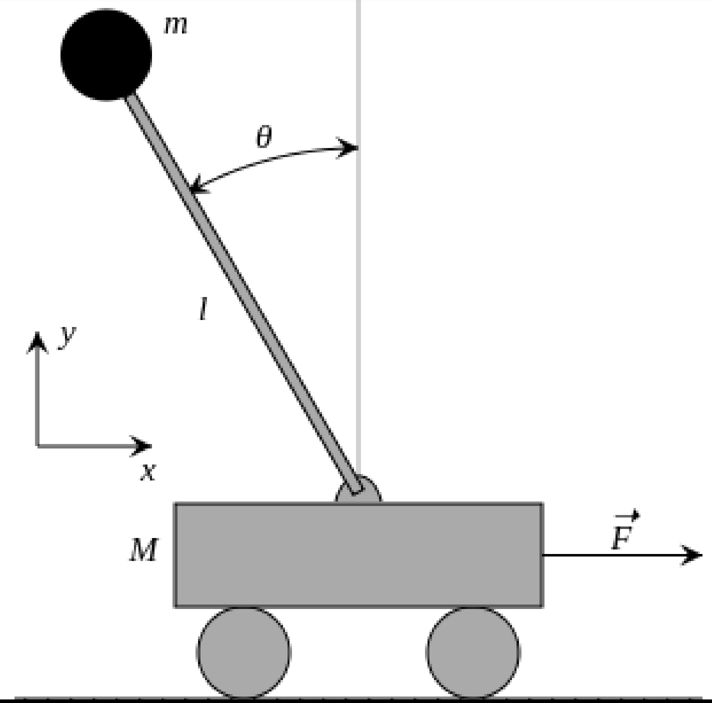


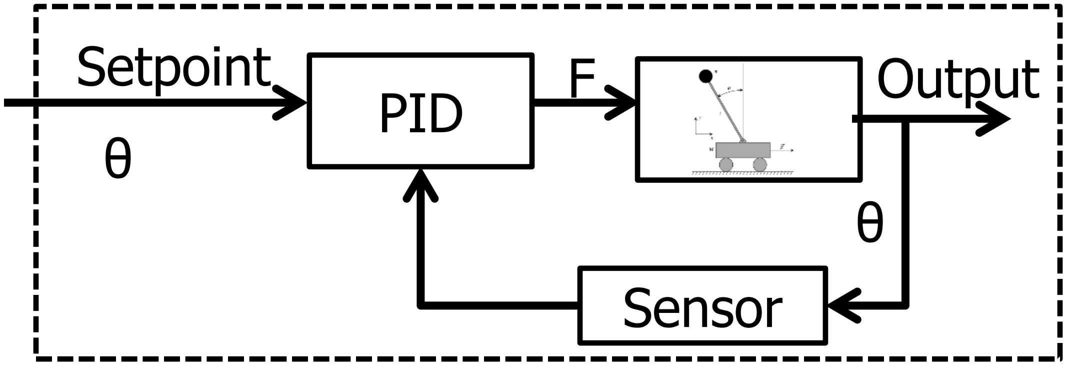
Inverted Pendulum (Nested PID)



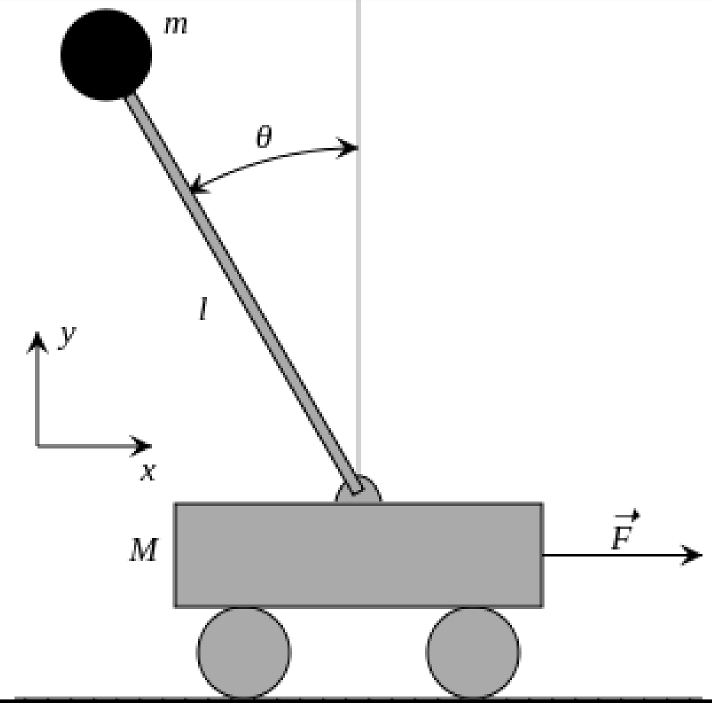


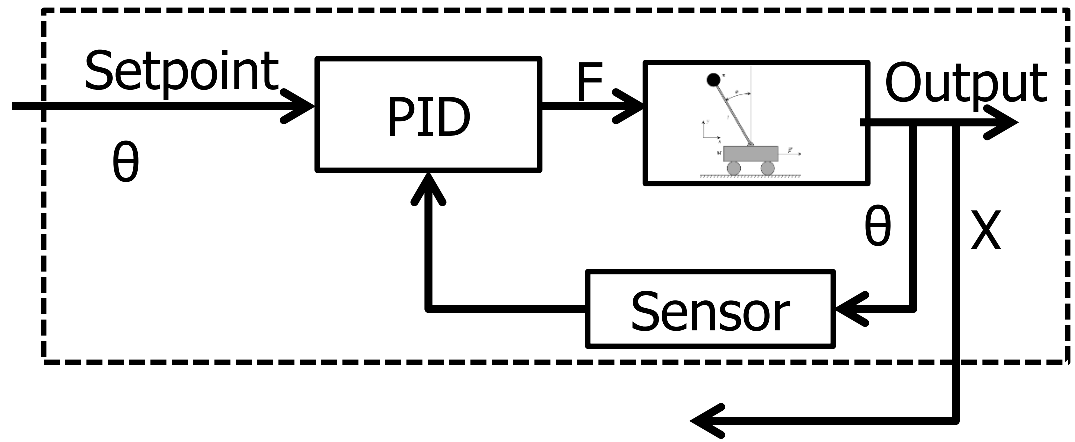
Nested PID



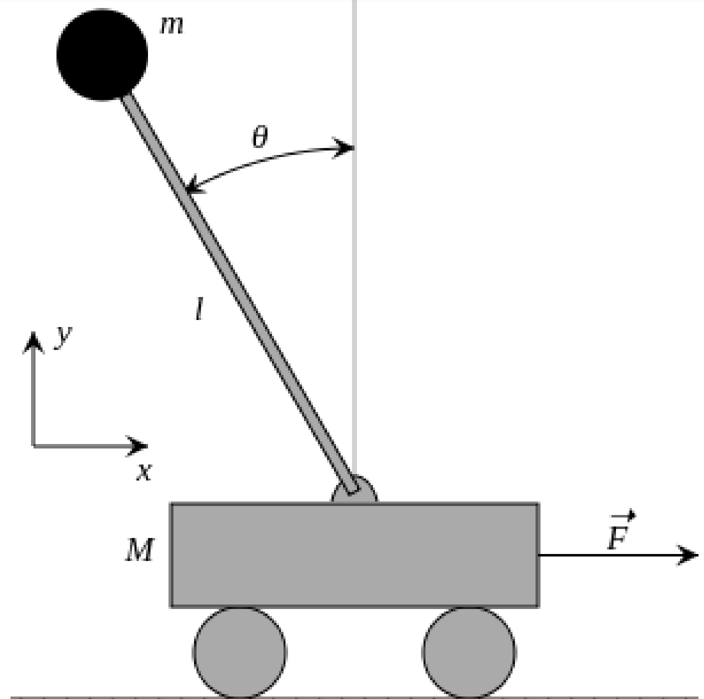


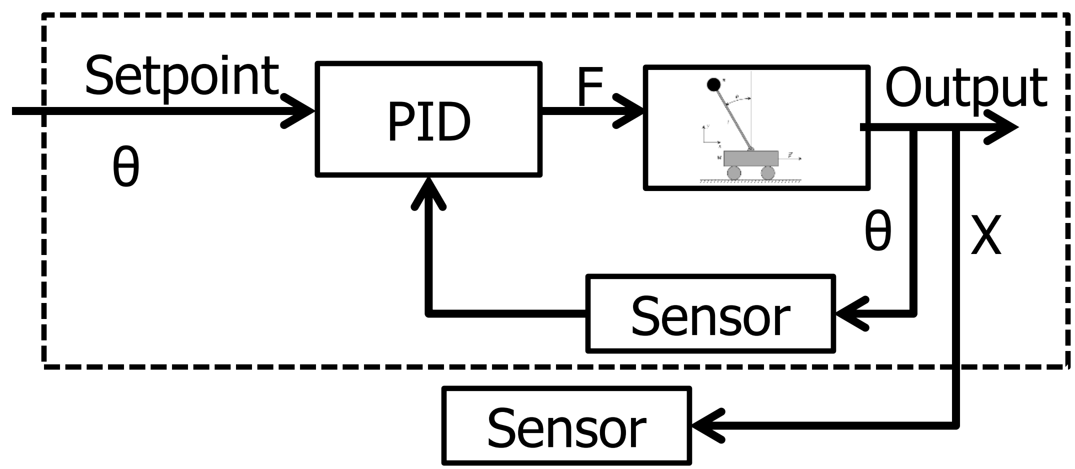
Nested PID (cont.)



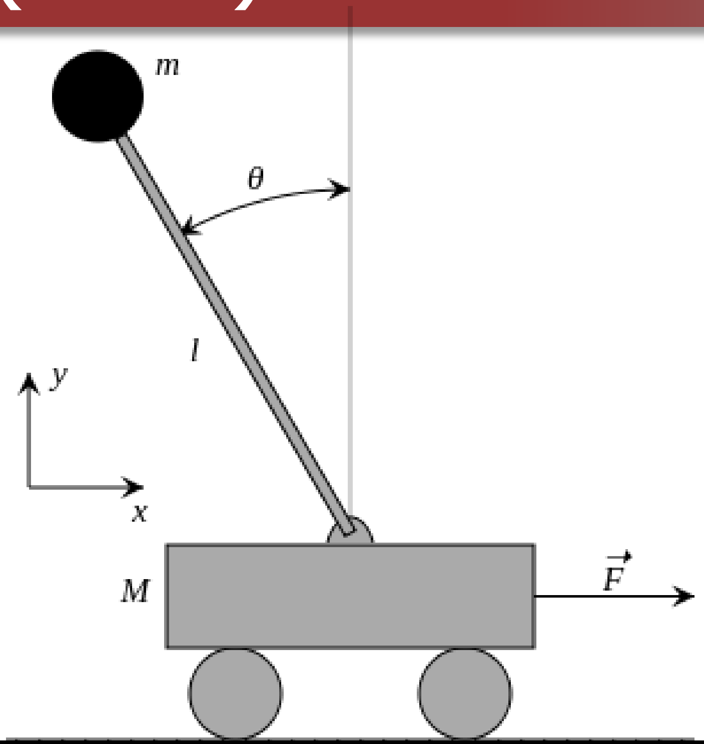


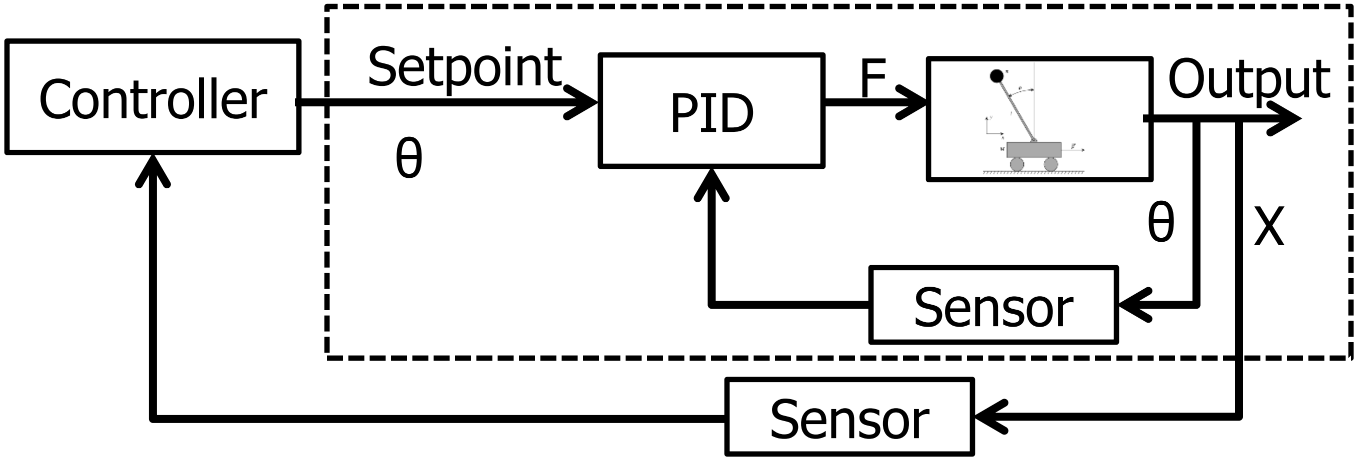
Nested PID (cont.)



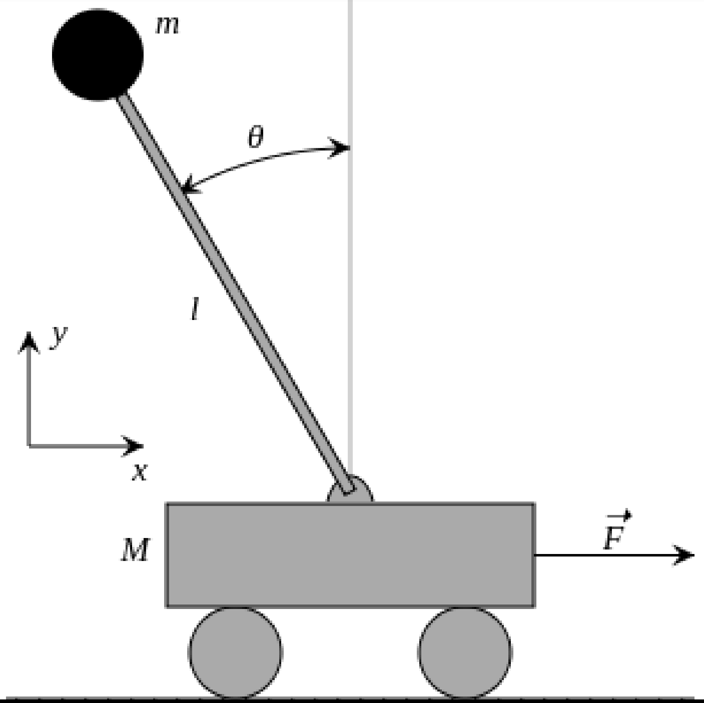


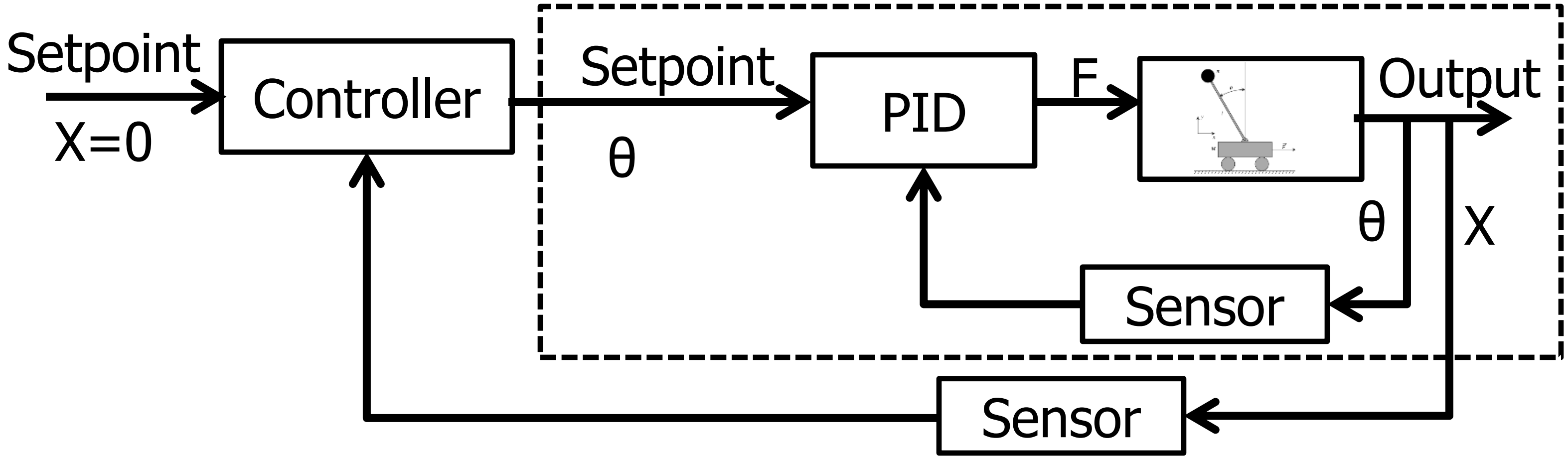
Nested PID (cont.)



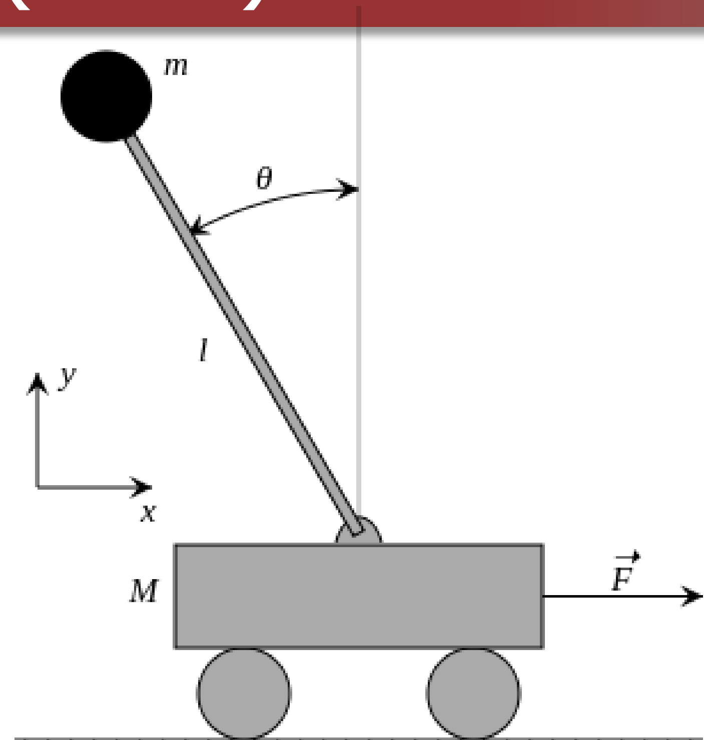


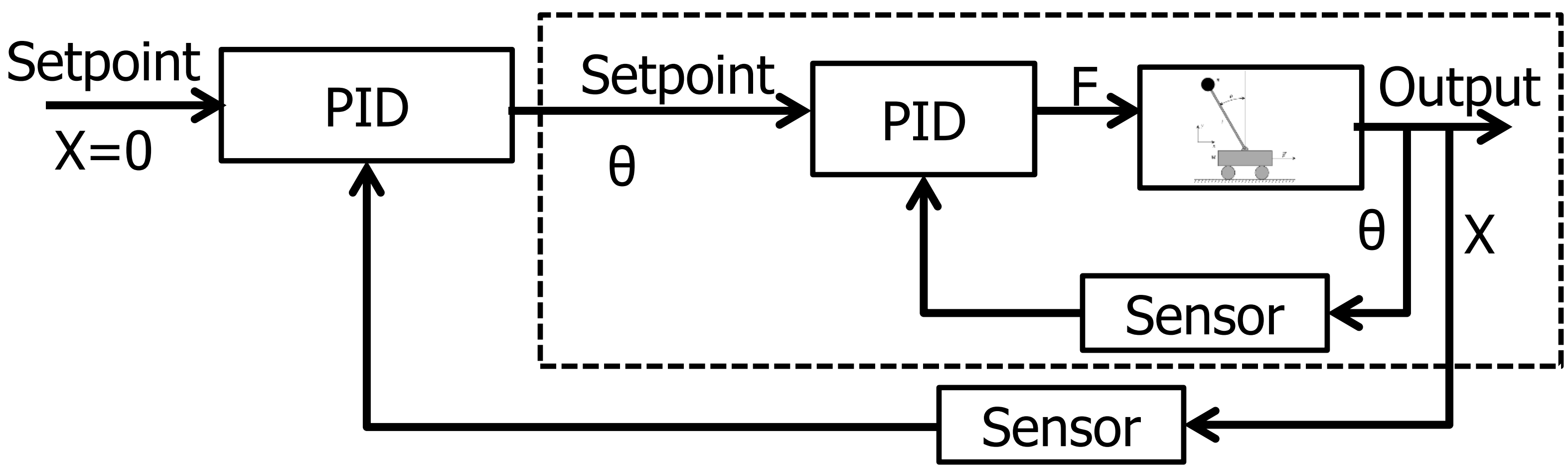
Nested PID (cont.)



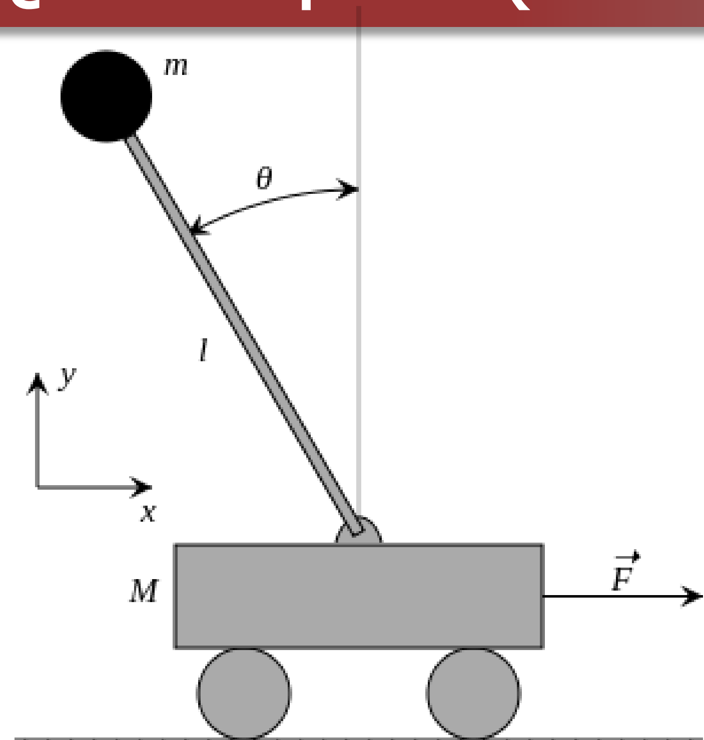


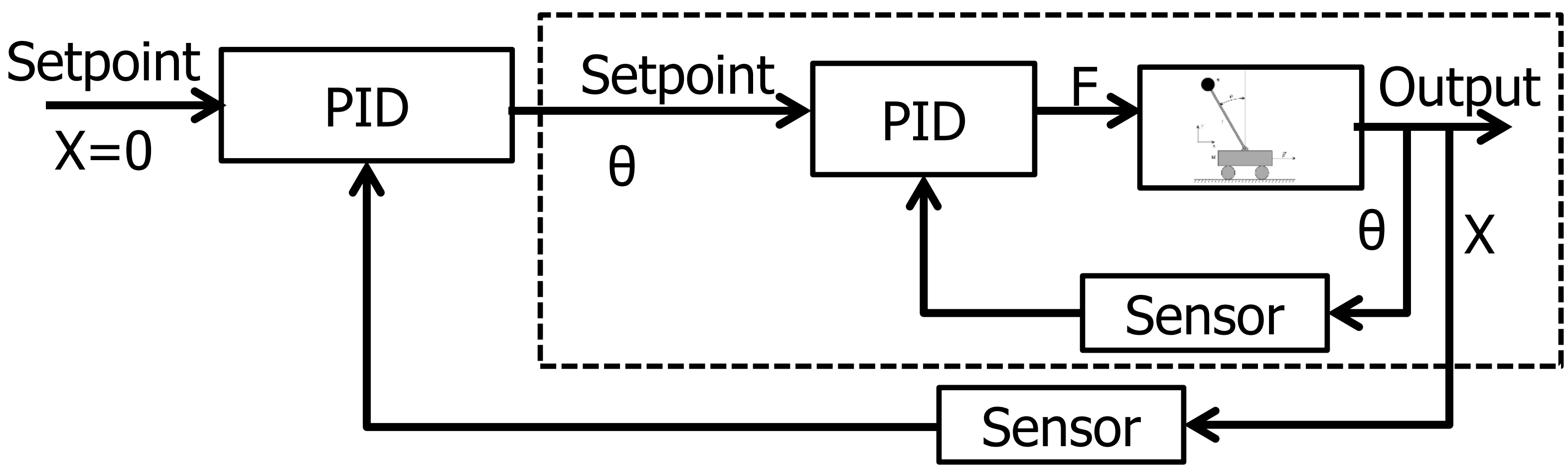
Nested PID (cont.)



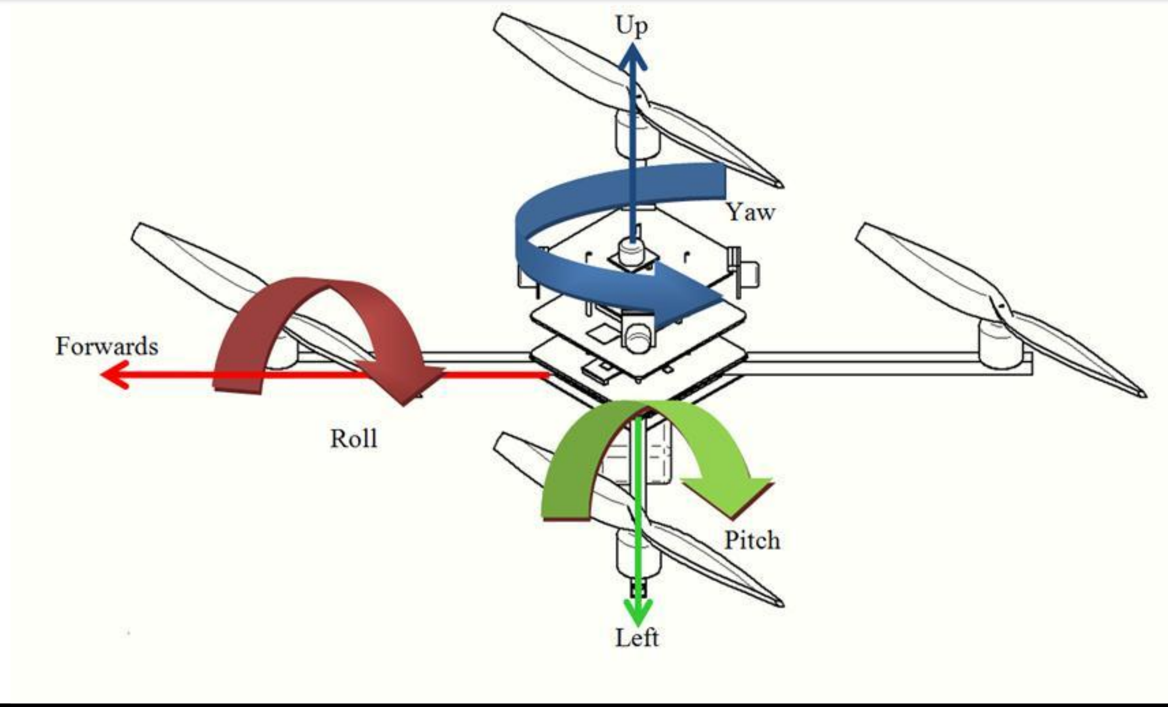


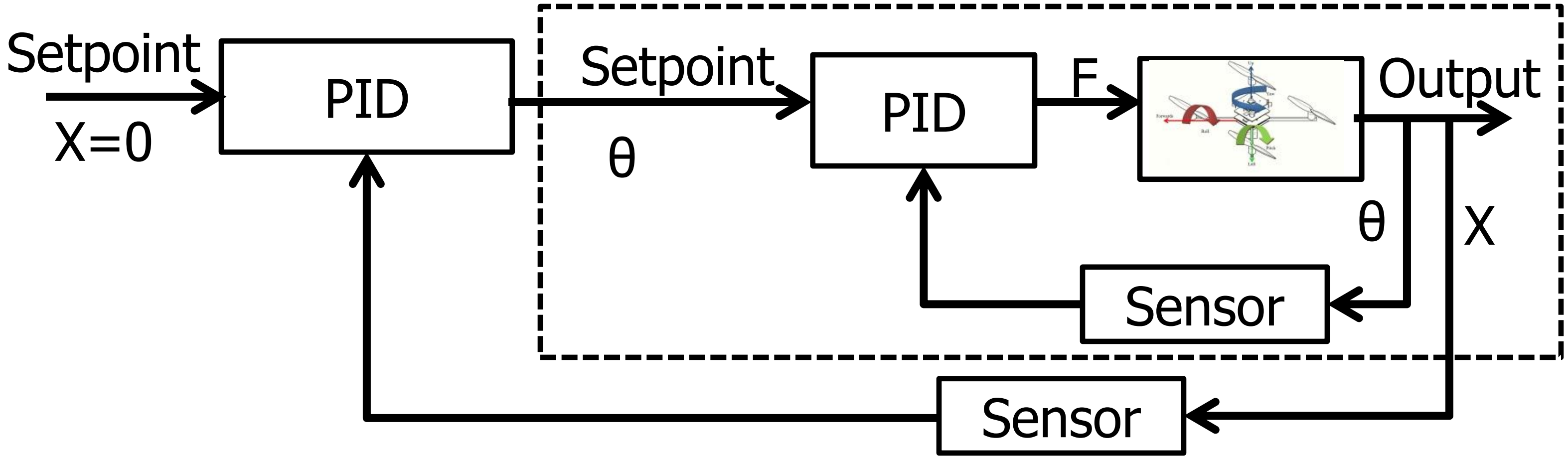
Relation to Quadcopter (Nested PID)





Relation to Quadcopter (Nested PID)





PID Tuning Techniques

- There are a few PID tuning techniques, more like rules of thumb (http://en.wikipedia.org/wiki/PID_controller)
  - Manual tuning
    1. Set KI and KD to 0 and increase KP until system oscillate, then turn down some
    2. Increase KI until steady state error is removed
    3. To reduces overshoot and settling time increase D
  - Ziegler–Nichols: heuristic method
    1. Set KI and KD to 0
    2. Based on the value of KP that causes the system to oscillate (i.e. KU) and the corresponding oscillation period (PU), KP, KI and KD are computed using the table below

Ziegler–Nichols method

Control Type	K_p	K_i	K_d
P	$0.50K_u$	-	-
PI	$0.45K_u$	$1.2K_p/P_u$	-
PID	$0.60K_u$	$2K_p/P_u$	$K_pP_u/8$

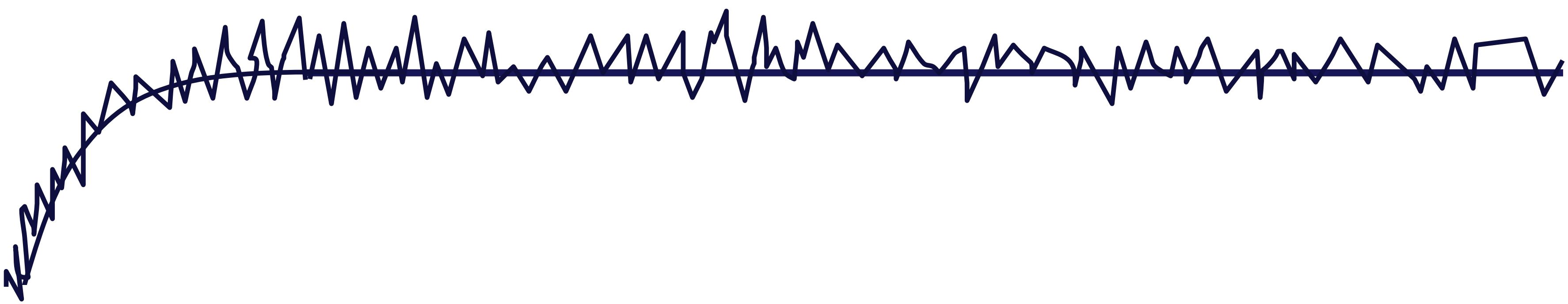
Revisiting the D Constant

- A large D constant will dampen the system, helping to keep it stable, but causing it to be slow in reacting.
- Are there any issues we need to be concerned with in a real system for a large D constant?

$$u[n] = K_P e[n] + K_I \sum_{j=0}^{n} e[j] + K_D(e[n] - e[n-1])$$

Revisiting the D Constant (cont.)

- A large D constant will dampen the system, helping to keep it stable, but causing it to be slow in reacting.
- Are there any issues we need to be concerned with in a real system for a large D constant?
- A large D constant will amplify the noise from the sensor which will case the controller to give large spikes of compensation.



$$u[n] = K_P e[n] + K_I \sum_{j=0}^{n} e[j] + K_D(e[n] - e[n-1])$$

Model-based Control

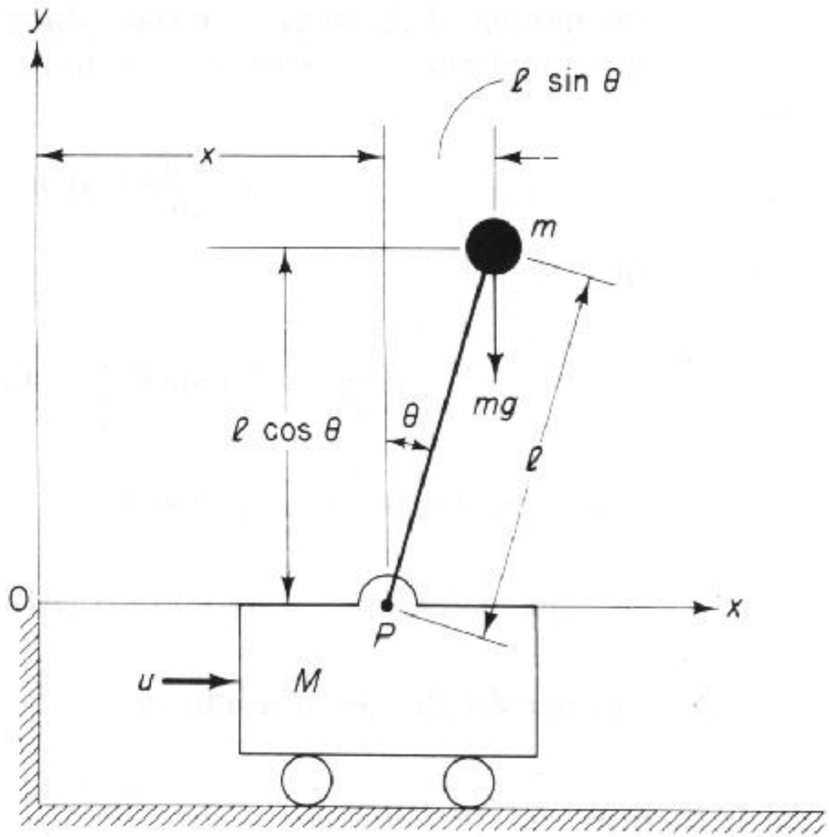
- Based on a model of the plant, mathematically design a controller
- Benefits?
- Draw backs?

Simple Car Model

- Velocity of car = x
- Acceleration of car = $\dot{x}$
- Mass of car = m
- Force acting on care = u (i.e. from gas petal)

$$\dot{x} = \frac{c}{m}u$$

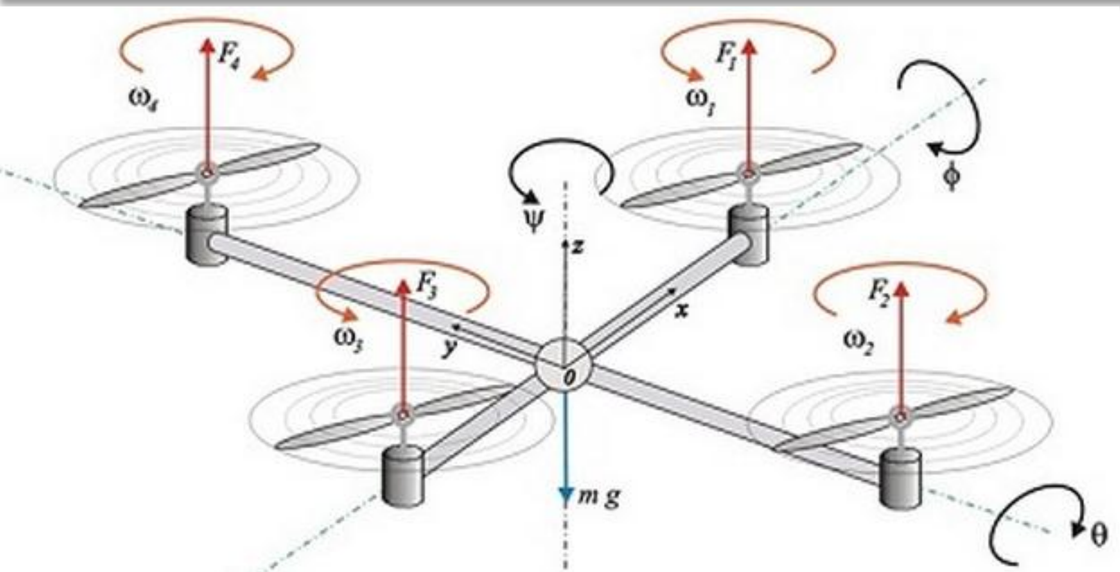
Inverted Pendulum Model



$$(M+m)\ddot{x} - m\ell\ddot{\theta}\cos\theta + m\ell\dot{\theta}^2\sin\theta = F$$

$$\ell\ddot{\theta} - g\sin\theta = \ddot{x}\cos\theta$$

Quadcopter Model



parameter	description	value
m	Mass of quadrotor	2.3 kg
L	Lever arm length	0.7 m
I_{xx}	Body inertia of x axis	8.04*10^-3 kgm^2
I_{yy}	Body inertia of y axis	8.46*10^-3 kgm^2
I_{zz}	Body inertia of z axis	14.68*10^-3 kgm^2
k_f	Force coefficient	0.65016e-3N/(rad/s)^2
k_t	Torque coefficient	0.82218e-5N/(rad/s)^2
k1	Motor model coefficient	20
k2	Motor model coefficient	0.01
k3	Motor model coefficient	3.5
z_l	Elastic deformation of gear	0.003 m
k_l	Elastic coefficient of gear	6000 N/m

$$\begin{bmatrix} \dot{p} \\ \dot{q} \\ \dot{r} \end{bmatrix} = -\begin{bmatrix} qr\left(I_{zz} - I_{yy}\right)/I_{xx} \\ pr\left(I_{xx} - I_{zz}\right)/I_{yy} \\ pq\left(I_{yy} - I_{xx}\right)/I_{zz} \end{bmatrix} + \begin{bmatrix} T_p/I_{xx} \\ T_q/I_{yy} \\ T_r/I_{zz} \end{bmatrix}$$

$$\begin{bmatrix} \dot{u} \\ \dot{v} \\ \dot{w} \end{bmatrix} = -\begin{bmatrix} -rv + qw \\ ru - pw \\ -qu + pv \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ F_z/m \end{bmatrix} + \begin{bmatrix} gs\theta \\ -gc\theta s\varphi \\ -gc\theta c\varphi \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ k_l\left(z_l - z\right)\delta\left(z_l - z\right)/m \end{bmatrix}$$

Model-based Control (cont.)

- Based on a model of the plant, mathematically design a controller
- State-space
  - x – state of system
  - Y – output
  - u – input
- Choose u to obtained desired y

$$\dot{x} = Ax + Bu$$
$$y = Cx$$

Model-based Control (cont.)

- Based on a model of the plant, mathematically design a controller.
- State-space
  - x – state of system
  - Y – output
  - u – input
- Choose u to obtained desired y

Matrix based off of the physics of the plant (i.e. math-model of the plant)

$$\dot{x} = Ax + Bu$$
$$y = Cx$$

Model-based Control (cont.)

- Based on a model of the plant, mathematically design a controller.
- State-space
  - x – state of system
  - Y – output
  - u – input
- Choose u to obtained desired y

Actuator matrix (i.e. math-model of how u gets translated into actuator commands)

$$\dot{x} = Ax + Bu$$
$$y = Cx$$

Model-based Control (cont.)

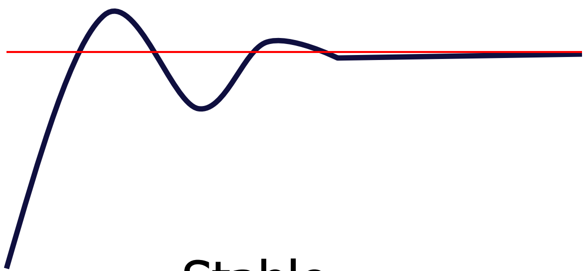
- Based on a model of the plant, mathematically design a controller.
- State-space
  - x – state of system
  - Y – output
  - u – input
- Choose u to obtained desired y

Sensor matrix (i.e. express what plant states you can observe with sensors)

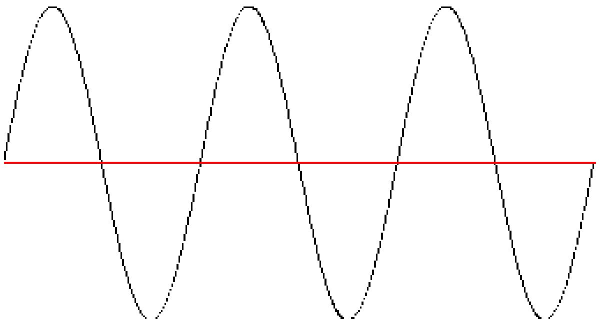
$$\dot{x} = Ax + Bu$$
$$y = Cx$$

General Controller Metrics

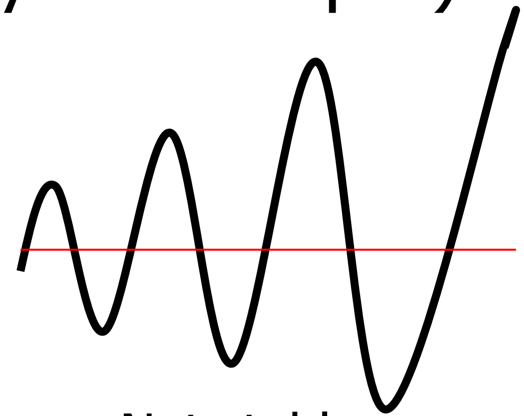
- Stability (e.g. bounded oscillation of system output)



Stable



Marginally stable



Not stable

- For a stable controller system
  - Disturbance Rejection: How well does system hold setpoint in the presence of a disturbance (e.g. shoving the quad on the turn table)
  - Command tracking: How well does the system respond to changes in the controller setpoint
    - Rise time
    - Settling time

Control Systems Summary

- PID (no plant model)
  - Benefits
    - Very useful for controlling many commonly found systems
    - Do not need to have much knowledge of the actual plant being controlled
  - Drawback
    - Only can control a single input single output (SISO)system
    - Can lead to hand tuning many constants.
    - Tuning even more challenging when dependencies
- PID (with plant model)
  - Benefit:
    - Easy to gain intuition for how constants impact system
    - There are tools that can computed constants (as a starting point)
  - Drawback:
    - If you have a plant model there are more advanced controllers that you can use (e.g. state space observer models)

Acknowledgments

- These slides are inspired in part by material developed and copyright by:
  - Maxim Raginsky (University of Illinois)
  - Magnus Egerstedt (Georgia Tech)