Cyber-Physical Systems

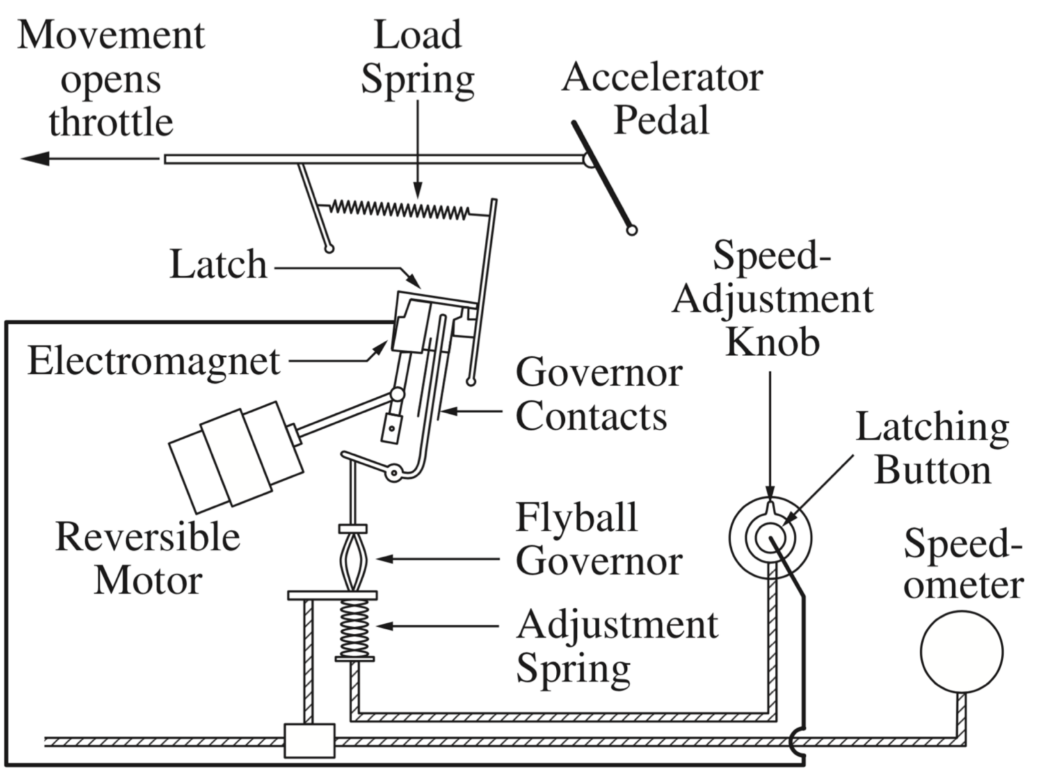
Feedback Control

ICEN 553/453– Fall 2018

Prof. Dola Saha

Control System in Action

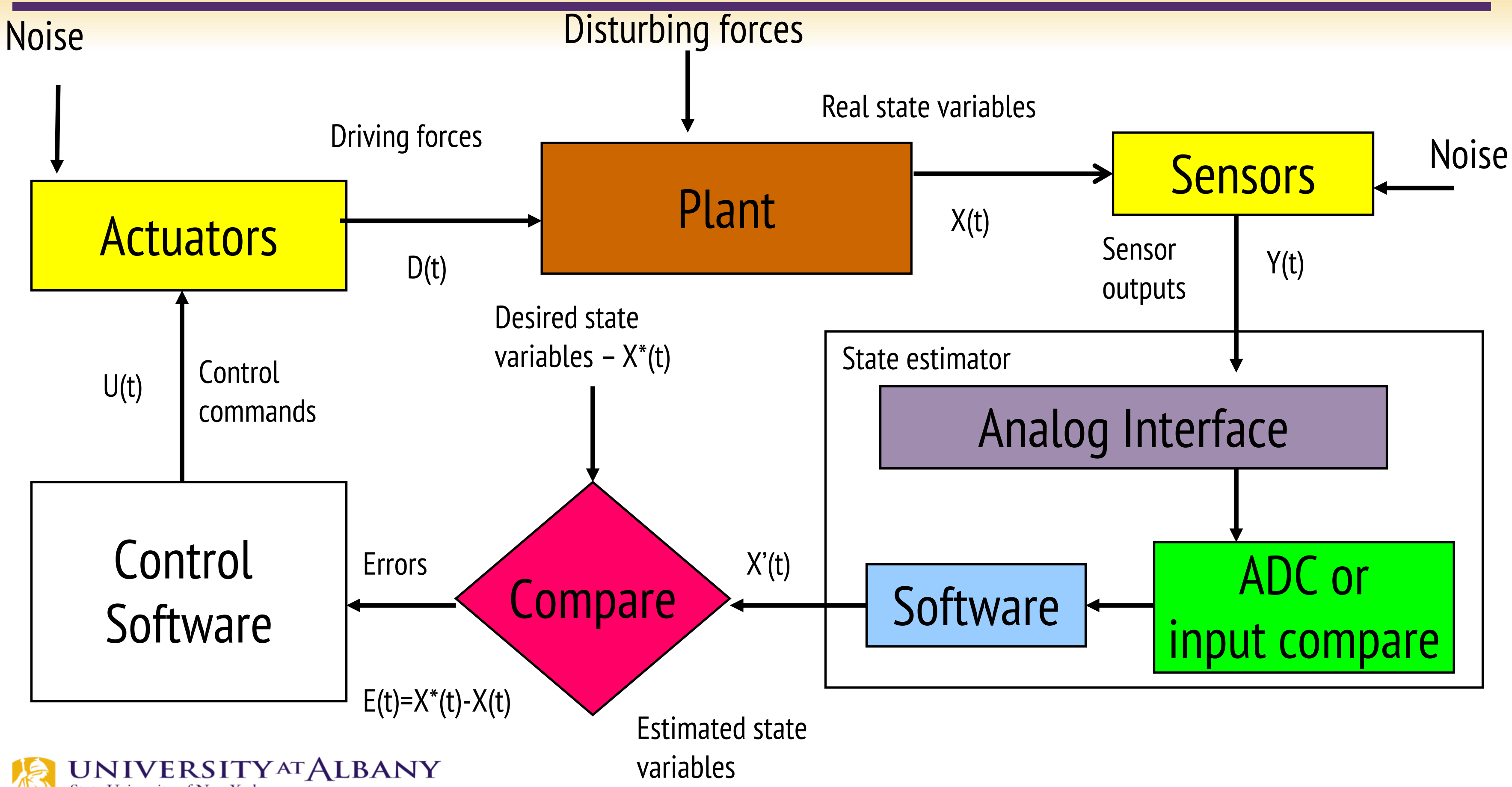
Honeywell Thermostat, 1953



Chrysler cruise control, 1958

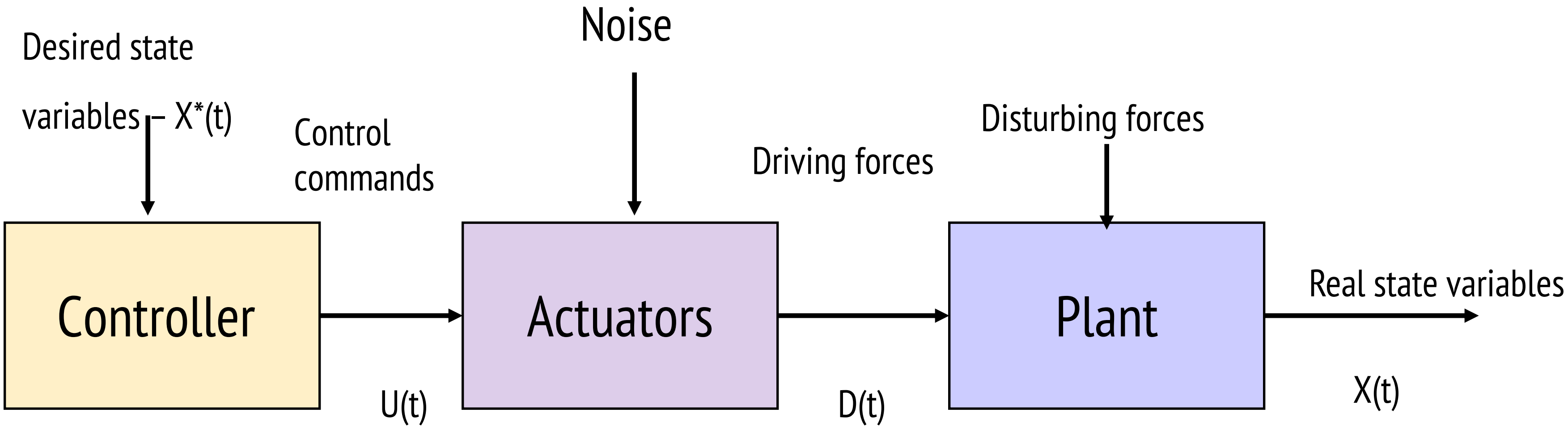
Feedback Systems: An Introduction for Scientists and Engineers

Closed Loop Control



Open Loop Control

- Control Action is independent of the output of the system



Open Loop Control

- state estimator eliminated
  - not well suited for a complex plant
- assumes disturbing forces have little effect on the plant
- less expensive than closed-loop control
  - example: electric toaster

Example Problem: Bike in straight Line

- Steer the bike in a straight line blindfolded
- Open loop → no sensor feedback
- What if you hit a rock?
- What if the handle bars aren't perpendicular to the wheels?

Control Systems Strategy

- Strategy
  - plant is a system that is intended to be controlled
  - collect information concerning the plant – data acquisition system (DAS)
  - compare with desired performance
  - generate outputs to bring plant closer to desired performance

- You can't control what you can't measure

Control Systems

- Microcomputers are widely employed in control systems:
  - automotive ABS, ignition and fuel systems
  - household appliances
  - smart things
  - industrial robots
  - pacemakers

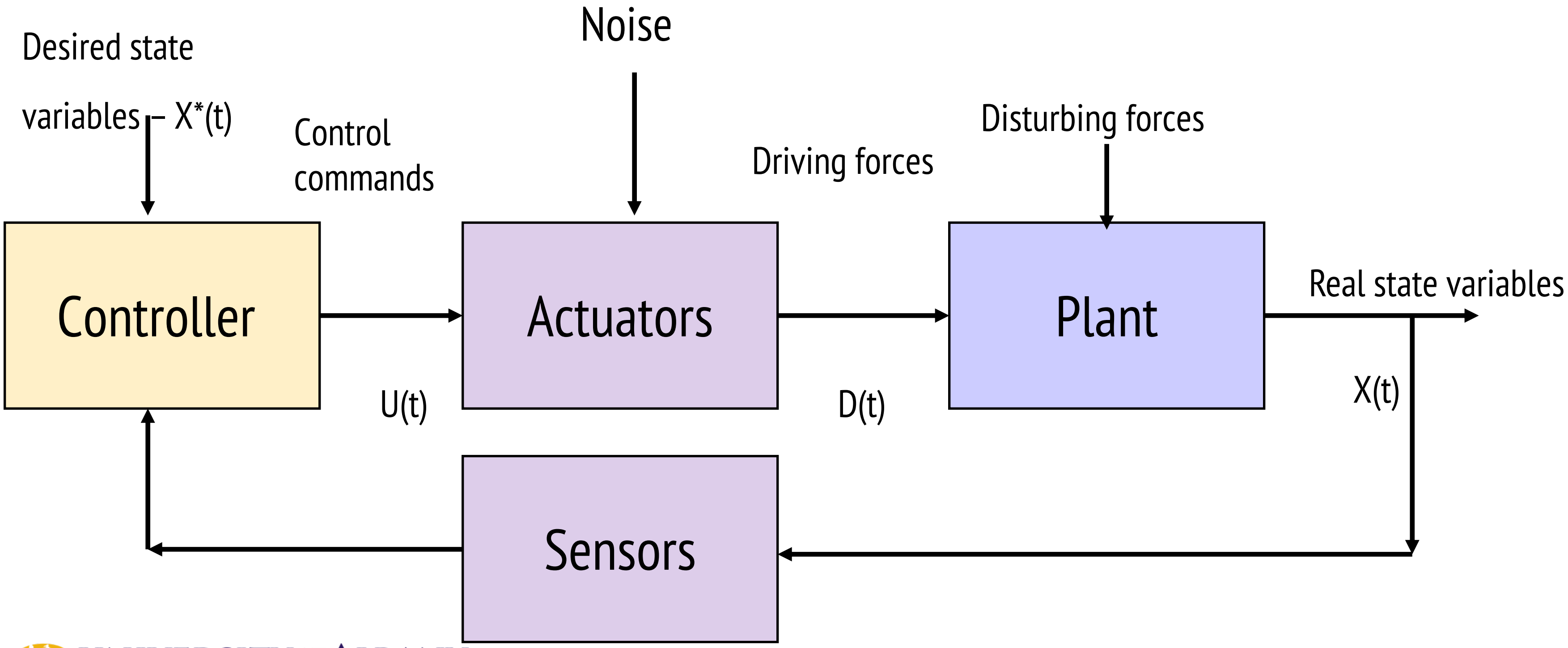
- Why are we interested in Feedback Systems in CPS course?

Control Systems – Closed loop

- Closed-loop control
  - feedback loop implementation
    - suitable for complex plant
  - sensors and state estimator produce representation/estimation of state variables
  - these values are compared to desired values
  - control software generates control commands based upon the differences between estimated and desired values

Closed Loop Control

- Control action depends on the output of the system

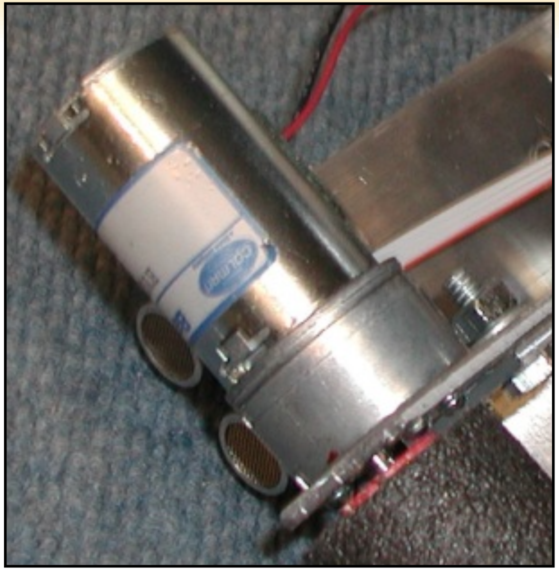


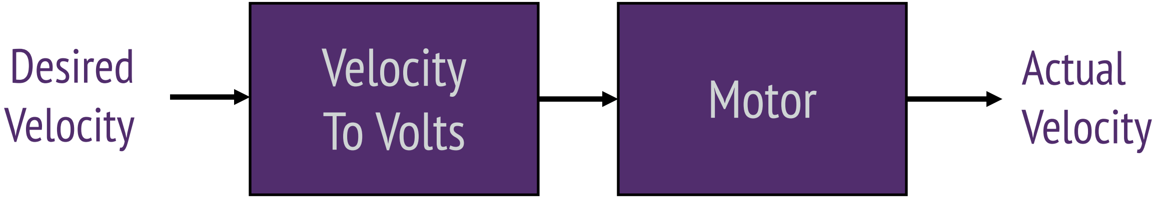
Example Problem: Bike in straight Line

- If you can see the pavement → Closed Loop Approach
- Control based on error: **PID**
- **Proportional** : Change handle angle proportional to the current error
- **Derivative** : Large handle corrections when error is changing slowly, and small handle corrections when error is changing quickly
- **Integral** : Handle corrections based on the cumulative error

Problem: Set Motor Velocity

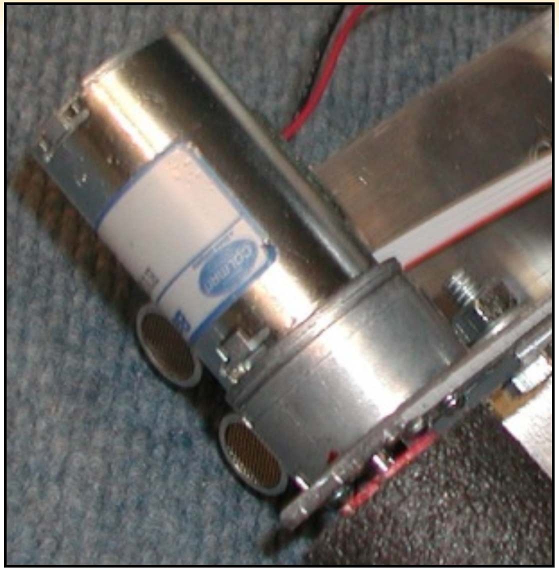
- Open Loop Controller
  - Use trial and error to create relationship between velocity and voltage
  - Problems
    - Supply voltage change
    - Bumps in carpet
    - Motor Transients

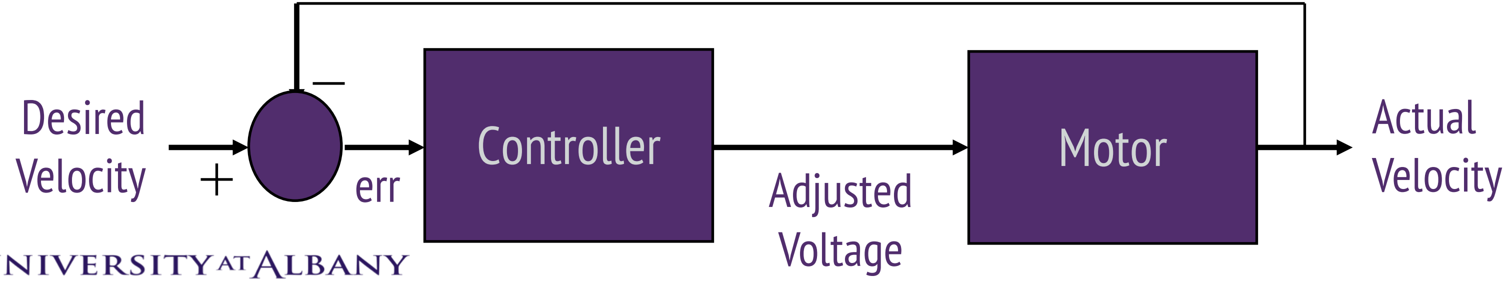




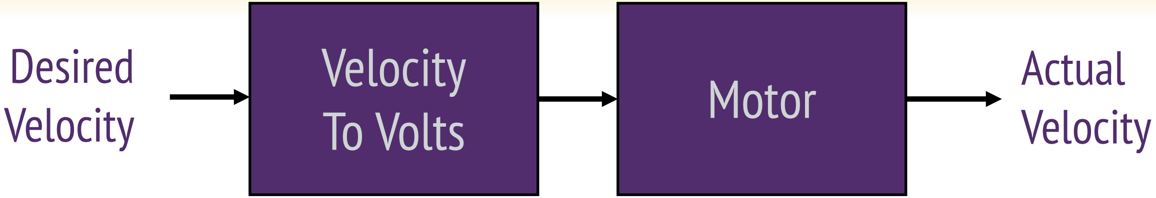
Problem: Set Motor Velocity

- Closed Loop Controller
  - Feedback is used so that the actual velocity equals the desired velocity
  - Can use an optical encoder to measure actual velocity

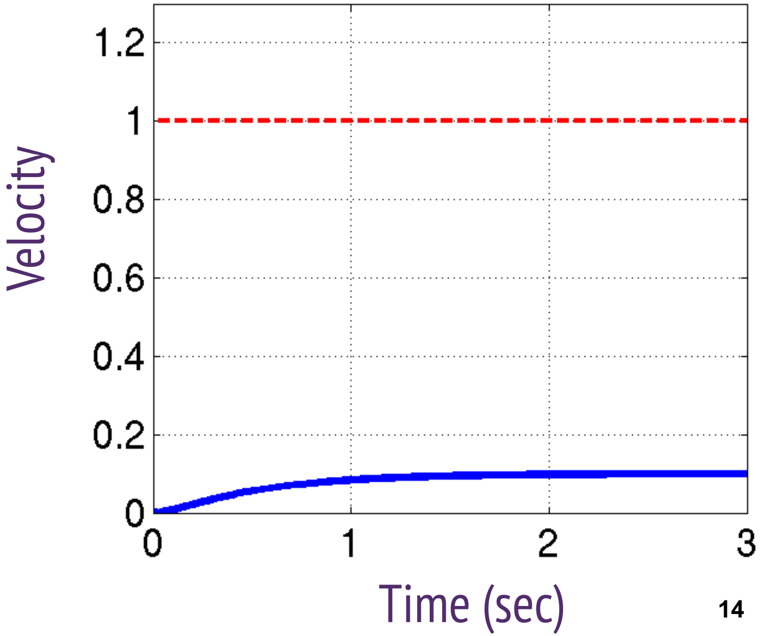




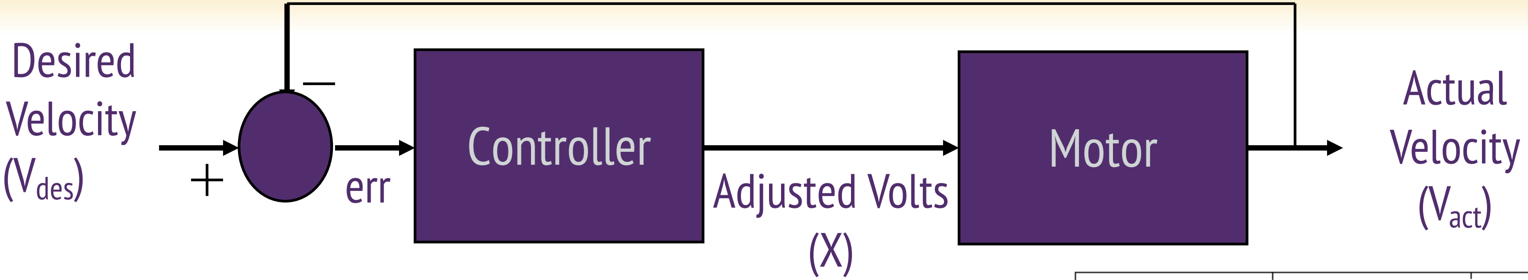
Step Response with No Controller



- Naive velocity to volts
- Model motor with several differential equations
- Slow rise time
- Stead-state offset

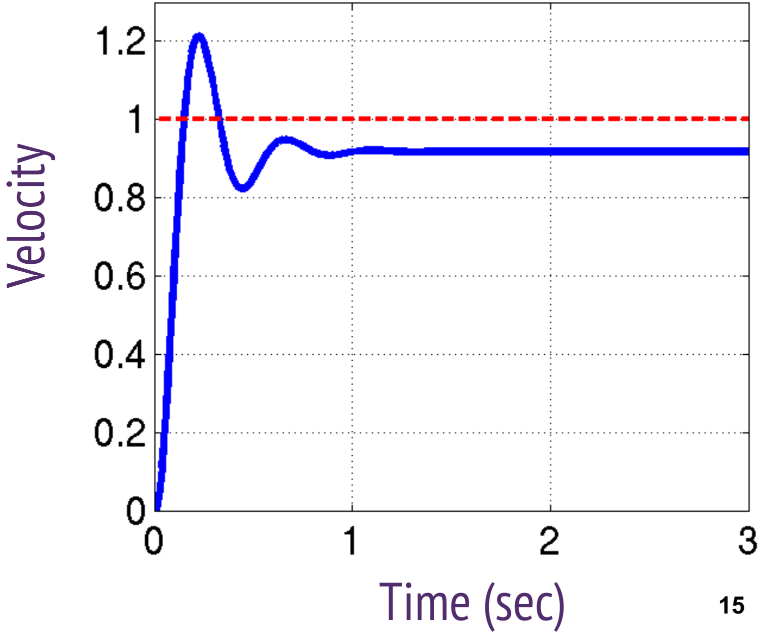


Step Response with Proportional Controller

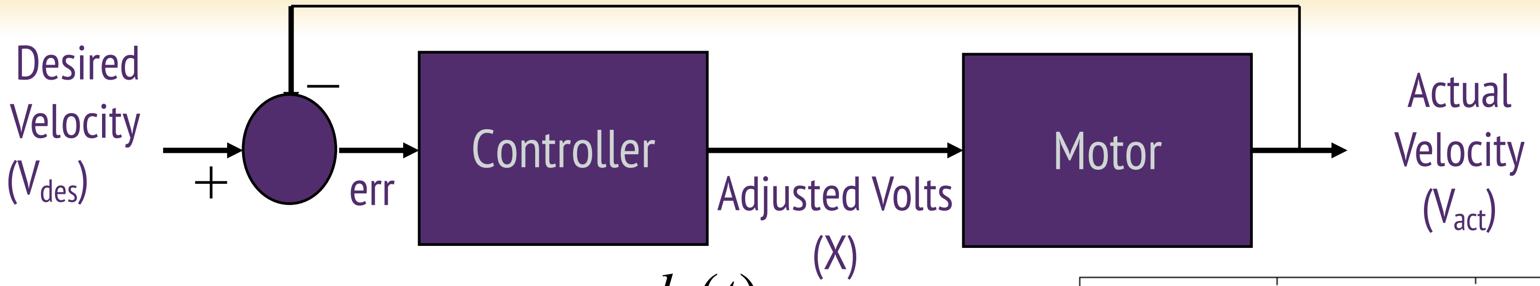


$$X = V_{des} + K_P \cdot (V_{des} - V_{act})$$

- Big error big = big adj
- Faster rise time
- Overshoot
- Stead-state offset (there is still an error but it is not changing!)

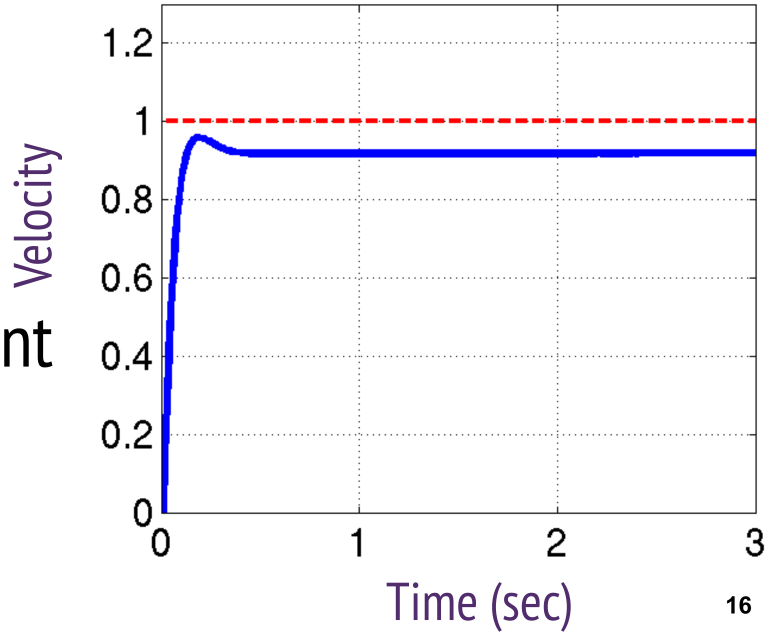


Step Response with PD Controller

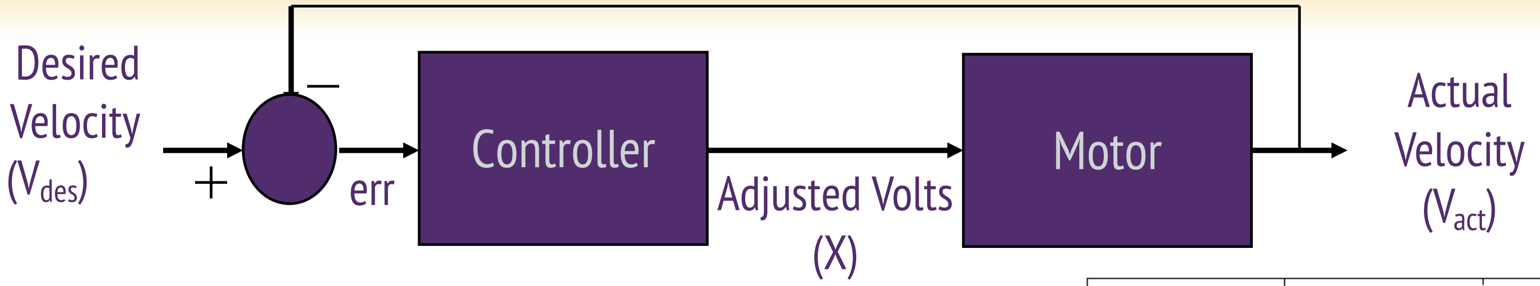


$$X = V_{des} + K_P e(t) - K_D \frac{de(t)}{dt}$$

- When approaching desired velocity quickly, de/dt term counteracts proportional term slowing adjustment
- Faster rise time
- Reduces overshoot

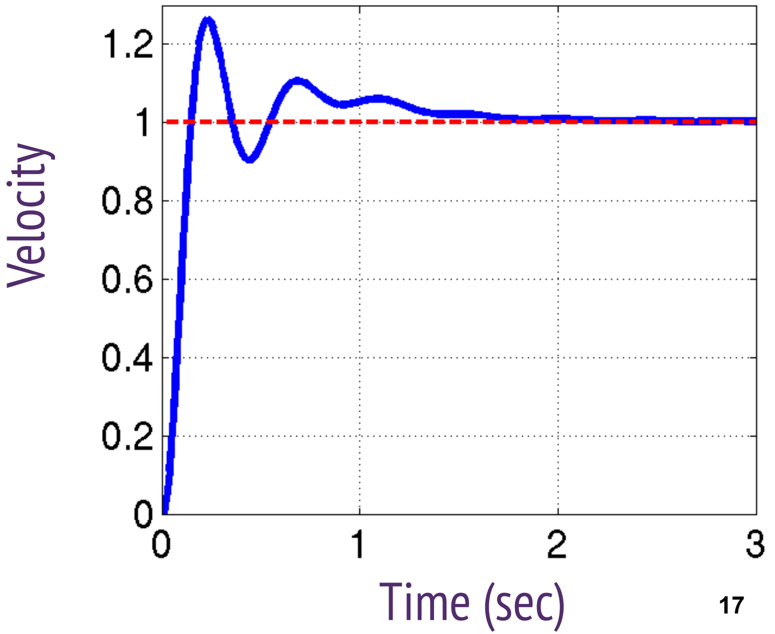


Step Response with PI Controller

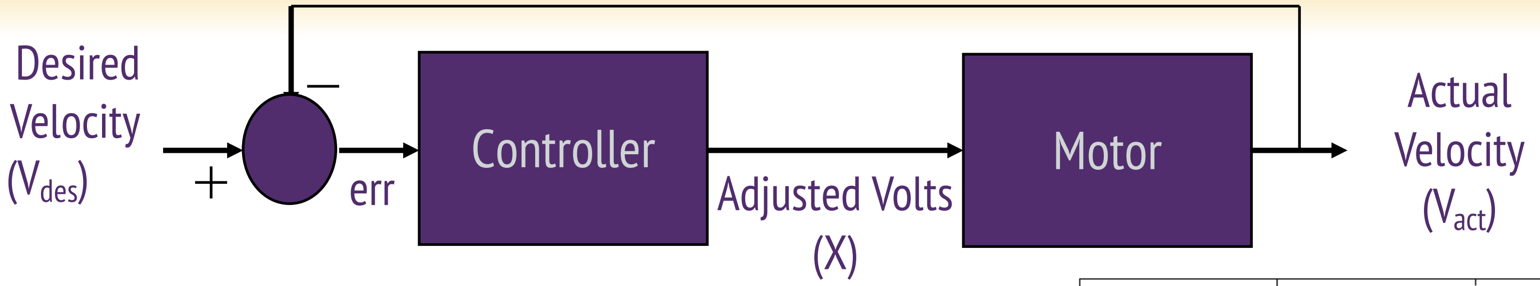


$$X = V_{des} + K_P e(t) - K_I \int e(t)\, dt$$

- Integral term eliminates accumulated error
- Increases overshoot

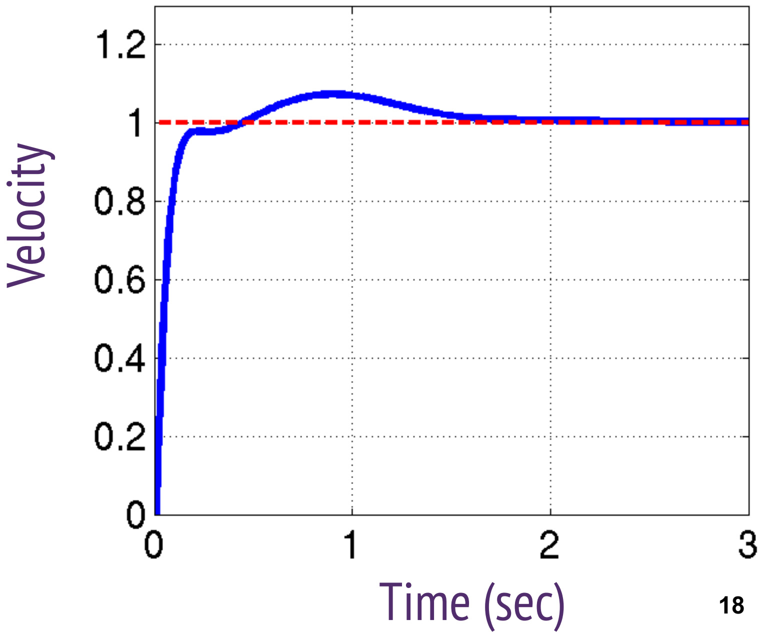


Step Response with PID Controller



- Combined benefits of PI and PD

$$X = V_{des} + K_P e(t) + K_I \int e(t)\, dt - K_D \frac{de(t)}{dt}$$



Control Systems – Performance

- Performance metrics
  - steady-state controller error
    - an average value of the difference between desired and actual performance
  - transient response
    - how quickly the system responds to change
  - stability
    - system output changes smoothly – without oscillation or unlimited excursions

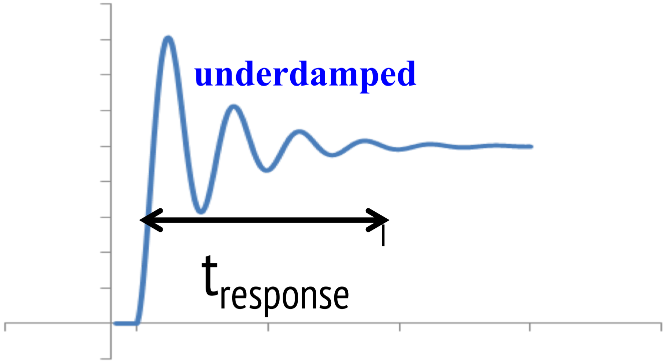
General Approach to PID

$$U(t) = K_p E(t) + \int_0^t K_i E(\tau) d\tau + K_d \frac{dE(t)}{dt}$$

- Proportional $U_p = K_p E$
- Integral $U_i = U_i + K_i \, E \, \Delta t$
- Derivative $U_d = K_d(E(n)-E(n-1))/ \Delta t$
- PID $U = U_p + U_i + U_d$

PID – Performance Measure

- Accuracy
  - Magnitude of the Error = Desired– Actual
- Stability
  - No oscillations
- Overshoot (underdamped, overdamped)
  - Ringing, slow
- Response Time to new steady state after
  - Change in desired setpoint
  - Change in load



Comparison

Controller	Response time	Overshoot	Error
Open-Loop	Smallest	Highest	Large
Proportional	Small	Large	Small
Integral	Decreases	Increases	Zero
Derivative	Increases	Decreases	Small change

Parameter Tuning

- Manual Tuning
- Ziegler–Nichols' Tuning
  - Time Domain Method
  - Frequency Domain Method
- Relay Feedback
- Integrator Windup

PID Controller in Software

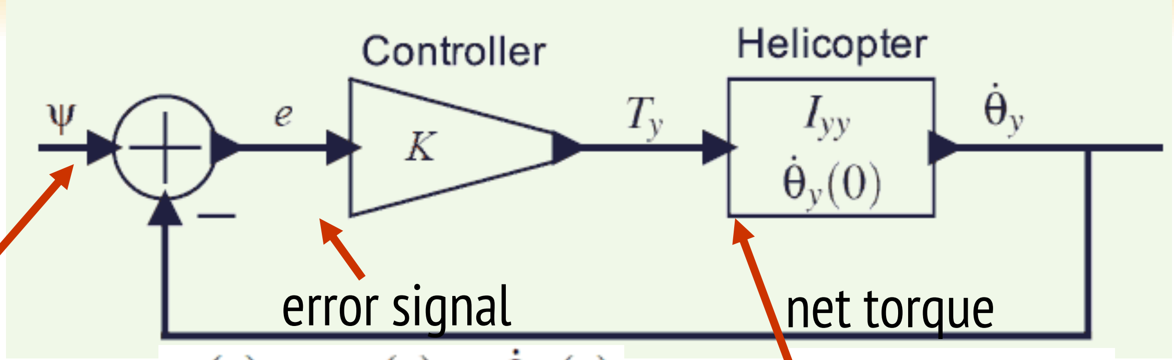
- Wait for clock interrupt
- Read input from sensor
- Compute control signal
- Send output to the actuator
- Update controller variables
- Repeat

PID Controller Pseudocode

```
% Precompute controller coefficients
bi=ki*h
ad=Tf/(Tf+h)
bd=kd/(Tf+h)
br=h/Tt

% Control algorithm - main loop
while (running) {
  r=adin(ch1)                 % read setpoint from ch1
  y=adin(ch2)                 % read process variable from ch2
  P=kp*(b*r-y)                % compute proportional part
  D=ad*D-bd*(y-yold)          % update derivative part
  v=P+I+D                     % compute temporary output
  u=sat(v,ulow,uhigh)         % simulate actuator saturation
  daout(ch1)                  % set analog output ch1
  I=I+bi*(r-y)+br*(u-v)       % update integral
  yold=y                      % update old process output
  sleep(h)                    % wait until next update interval
}
```

Proportional Controller to Helicopter Problem



desired angular velocity

error signal

$$e(t) = \psi(t) - \dot{\theta}_y(t)$$

net torque

$$T_y(t) = Ke(t)$$

$$\dot{\theta}_y(t) = \dot{\theta}_y(0) + \frac{1}{I_{yy}} \int_0^t T_y(\tau)d\tau$$

$$= \dot{\theta}_y(0) + \frac{K}{I_{yy}} \int_0^t (\psi(\tau) - \dot{\theta}_y(\tau))d\tau$$

Controller Summary

- Controller only as good as its sensor
- Observe everything "What was it thinking?"
- Change one parameter at a time
- Choose stability over responsiveness