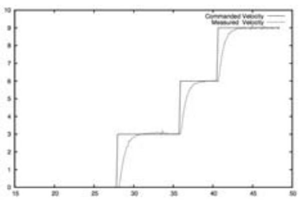
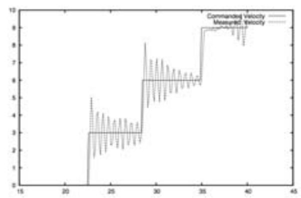


Motor Control



RSS Lecture 2

Monday, 10 Feb 2014

Prof. Seth Teller

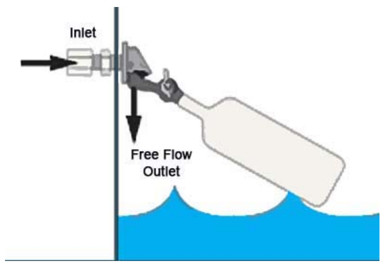
Jones, Flynn & Seiger § 7.8.2

Today: Control

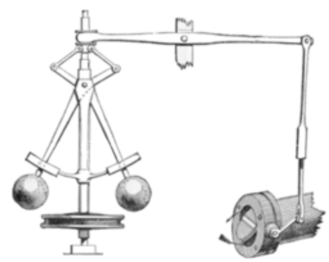
- Early mechanical examples
- Feed-forward and Feedback control
- Terminology
- Basic controllers:
  - Feed-Forward (FF) control
  - Bang-Bang control
  - Proportional (P) control
  - The D term: Proportional-Derivative (PD) control
  - The I term: Proportional-Integral (PI) control
  - Proportional-Integral-Derivative (PID) control
- Gain selection
- Applications

What is the point of control?

- Consider any mechanism with adjustable DOFs* (e.g. a valve, furnace, engine, car, robot…)
- Control is *purposeful variation* of these DOFs to achieve some specified *maintenance state*
  - Early mechanical examples:† ,



www.freshwatersystems.com



wikimedia commons

*DOFs = Degrees of Freedom

†Note blanks on your printed slides!

Water clock due to Archimedes (c. 300 B.C.)

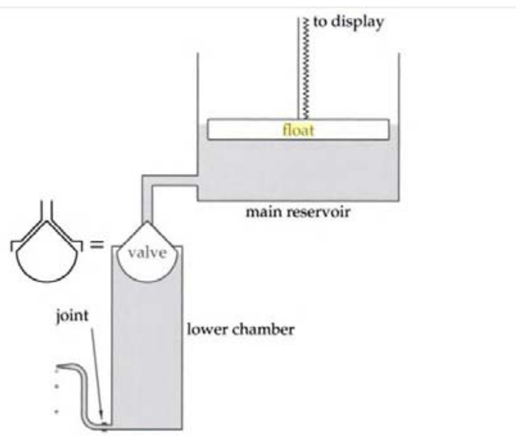


FIGURE 4.5. Water clock attributed to Archimedes. The pressure at the bottom of the lower chamber is kept constant by means of a float valve. A joint lets the output pipe be directed vertically, horizontally or diagonally, thus allowing for seasonal adjustments in the flow rate to compensate for the variable length of the hour (the rate is controlled by the difference between the water level in the lower chamber and the height of the outer spout).

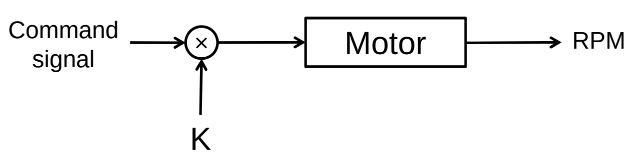
Lucio Russo, The forgotten revolution: how science was born in 300 BC and why it had to be reborn, Birkhauser 2004

The Role of Control

- Many robotics tasks are defined by (high-level) *achievement goals* requiring *planning*, e.g.:
  - Go to the exit of the maze
  - Push a box around some obstacles to a goal location
  - Pass to the left around a slower-moving vehicle
- Other robotics tasks are defined by (low-level) *maintenance goals* requiring *control*, e.g.:
  - Drive at 60 mph (or in RSS, roll forward at 0.5 m/s!)
  - Keep to the center of the lane indefinitely
  - Follow some trajectory computed by the planner
  - Balance on one leg
- Today's focus is control; we'll see planning later

Feed-Forward (FF) Control

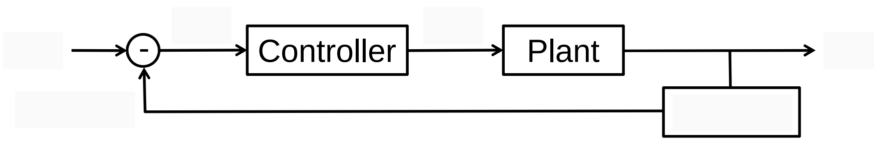
- Pass command signal from external environment directly to the *loaded element* (e.g., the motor)
- Command signal typically multiplied by a *gain* K



- … What are the *units* of the command signal?
- … Where does the gain value K come from?
  –
- Under what conditions will FF control work well?
  –
- You will implement a FF controller in Lab

Feedback Control Terminology

- *Plant* **P**: process commanded by a *Controller*
- *Process Variable* **PV**: Value of some process or system quantity of interest (e.g. temperature, speed, force, …) as measured by a *Sensor*
- *Set Point** **SP**: Desired value of that quantity

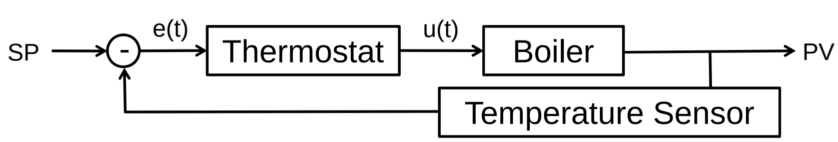


- *Error signal* **e(t)** = error in the process variable at time t, computed via
- *Control signal* **u(t)**: controller output (value of switch, voltage, PWM, throttle, steer angle, …)

*Set point is sometimes called the "Reference"

Example: Home Heating System

- *Plant* P: Boiler with on-off switch (1 = all on ; 0 = all off)
- *Process Variable* PV:
- *Controller:* *Sensor:*
- *Set Point* SP:
- *Control signal*:

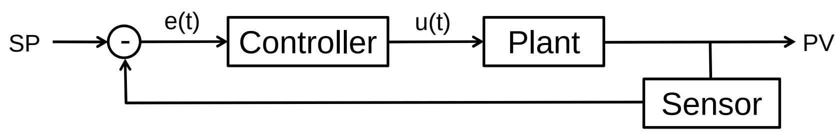


How could the function **u(t)** be implemented?

This is called "**bang-bang control**." Would it work well?

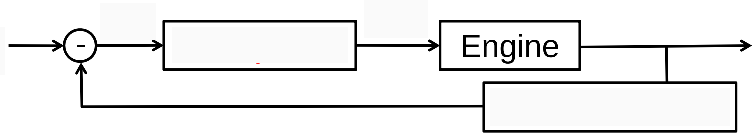
Proportional Control

- Suppose plant can be commanded by a *continuous*, rather than discrete, signal, e.g.:
  - Valve position to a pipeline or carburetor
  - Throttle to an internal combustion engine
  - PWM value to a DC motor
- What's a natural thing to try?
  - *Proportional* (P) C*ontrol*: make the command signal



Example: Cruise Control (CC) System

- *Plant* P: Engine with throttle setting u ∈ [0..1]
- *Process Variable* PV:
- *Controller:* *Sensor:*
- *Set Point* SP:
- *Control signal*:



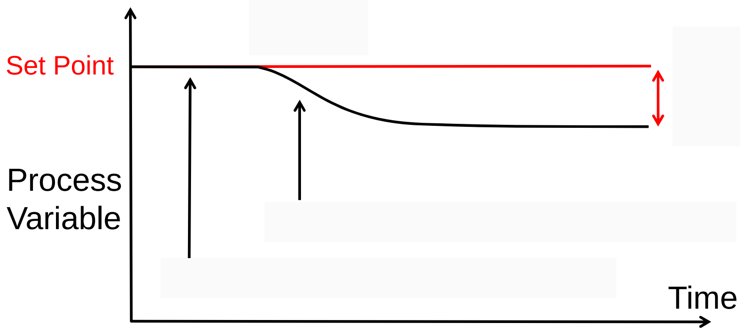
Define e(t) = , u(t) = ,

i.e. Throttle =

Does this controller "settle" at the desired speed?

Proportional Control: Why SSE?

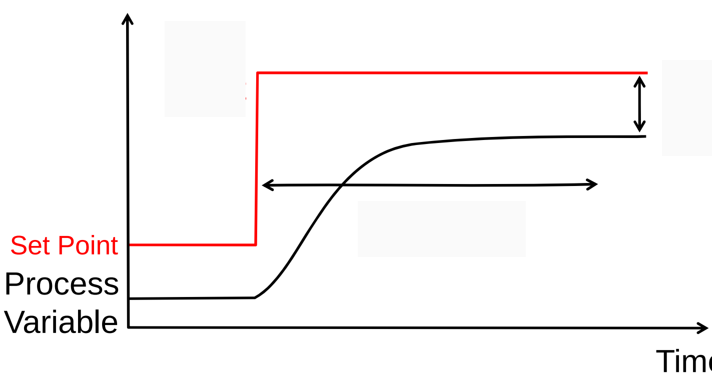
- Suppose PV=SP. Then u(t) =
- Process Variable
- But any real physical system has a
- Deviation,



Why not just introduce constant term, $u(t) = A + K_P * e(t)$?

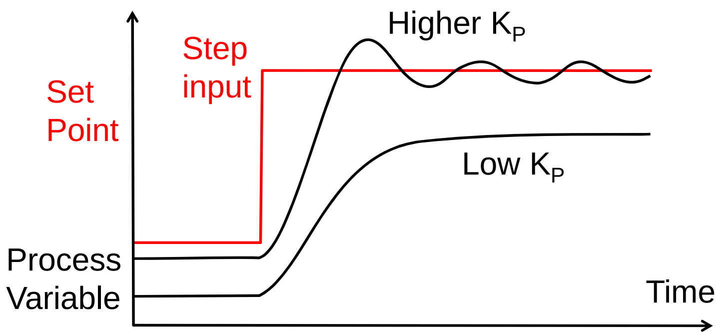
Proportional Control Step Response

Notional plot and terminology:



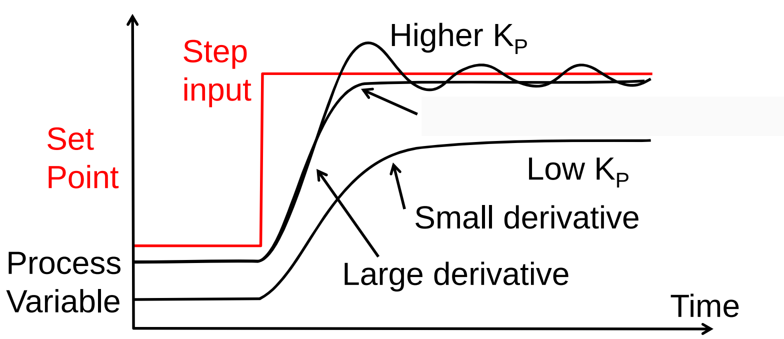
Proportional Control and SSE

- Can combat SSE by (“the P gain”)
- This gives a and
- But leads to



Combatting Overshoot: The D Term

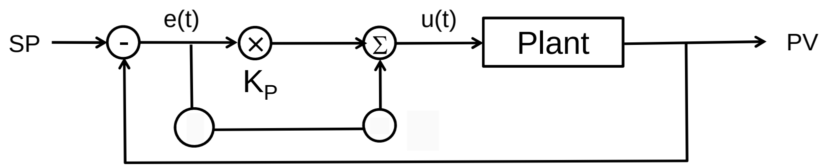
- Note the *derivative* of error in responses below
- it from output to counteract overshoot
- Then $u(t) = K_P \times e(t)$
  - Adds “derivative” or “damping” term to PD controller



- … But still haven’t eliminated steady-state error!

Combatting Steady-State Error: I Term

- Idea: apply correction based on *integrated* error
  - If error persists, *integrated* term will grow in magnitude
  - Sum proportional and integral term into control output

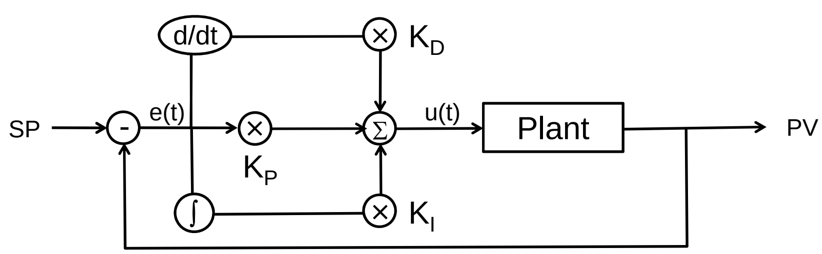


Then $u(t) = K_P \times e(t) +$ (where the integral of the error term is taken over some specified time interval)
This produces a *proportional-integral* (PI) controller

Incorporating the I term eliminates SSE by modulating the plant input so that the .
You'll hear robotics people speak of controller "wind-up"

Putting it All Together: PID Control

- Incorporate P, I and D terms in controller output
  - Combine as a weighted sum, using gains as weights



Then $u(t) =$ + +
This is a "proportional-integral-derivative" or *PID controller*
(In "ideal form," controller gains are physically meaningful)

Are gains *unitless*? …. interpretable as *physical quantities*?

How to determine effective gain values?

- P controller: search 1-D space of gains K_P
  - Identify various behavior regimes; you'll do this in Lab
- Choose analytical or empirical approach (how?)
- Hybrid: Ziegler-Nichols Tuning Method (Heuristic)
  - Useful in absence of system model (if system)
  - Start with pure P control (how?); Increase K_P until system *oscillates indefinitely*; note critical gain K_C and period T_C
  - Then for P, PI, or PID control, set gains as follows:

	K_P	K_I	K_D
P	$0.5\ K_C$		
PI	$0.45\ K_C$	$1.2\ K_P / T_C$	
PID	$0.6\ K_C$	$2\ K_P / T_C$	$K_P T_C / 8$

  - Yields acceptable but not optimal controller behavior

Other Applications of Feedback Control

- Mobility:
  - Lane-keeping
  - Trajectory-following
  - Standoff maintenance
- Manipulation:
  - Maintaining a steady contact force for grasping
  - Holding a mass at a certain location or attitude
  - Pushing a sliding object at constant velocity
- Sensing:
  - Automatic gain control, white balance, etc.
  - Target-tracking for active vision (body, head, eyes…)
- Many, many more

To Think About

- Lab 2 involves running motor at constant speed
- Lab 3 involves following a hand-held ball
- Lab 4 involves moving alongside a solid wall
- Lab 6 involves picking up a block from the ground
- How might you use P/I/D feedback control to implement any of these behaviors?
- What sensor(s) would you use, and what sort of error signal(s) would you infer from them?
- What would your robot's behavior look like?

What's Next?

- For more on control, consider taking any of:
  - 2.003, 2.004, 2.086, 2.12, 2.14x, 2.151, 2.152, 2.830, …
  - 6.01, 6.003, 6.011, 6.142, 6.231, 6.241, 6.243, 6.832, …
  - 16.06, 16.30, 16.31, 16.301, 16.32x, 16.72 (ATC!), …
  - 9.05, 9.272, 10.450, 10.976, HST.545, …

- Today & Wed in Lab: implementing controllers
- Wednesday lecture: Electric motors
- Lab 2 wiki materials, briefings next Wed 19 Feb
  - We'll cover expectations for briefings in F 2/14 Forum