Lecture #22

Introduction To Embedded Control

18-348 Embedded System Engineering

Philip Koopman

Wednesday, 6-April-2016

Electrical & Computer ENGINEERING

Carnegie Mellon

Where Are We Now?

- **Where we've been:**
  - Lots and lots of places ..
  - Last lecture we started cleaning up loose ends and taking a quick look at advanced topics
    -
- **Where we're going today:**
  - Lightweight introduction to control theory
    -
- **Where we're going next:**
  - Real Time Operating Systems
  - Bluetooth, CAN, networking
  - Booting, Power Management & Robust Systems
  - Exam #2 is Wednesday April 20, 2016
  - Final projects

Rice Cooker Revenues Top Small Kitchen Appliance Ranking

Written by IMS Research

Rice Cooker Revenues Top Small Kitchen Appliance Ranking - 27 September 2006

Wellingborough, UK (September 25, 2006) — Rice cookers were the number one revenue contributor in 2005 within the worldwide small kitchen appliance market, followed by kettles, coffee makers and blenders, according to the latest report from IMS Research entitled The Worldwide Market for Small Household Appliances.

The study reveals that the largest small kitchen appliance type, in terms of volumes and revenues, in 2005 was rice cookers, with nearly 40 million units shipped worldwide, generating more than $1 billion of revenue. Kettles, drip-filter coffee makers, espresso coffee makers and blenders each generated more than $650 million worldwide in 2005. The worldwide rice cooker market is forecast to reach nearly 50 million units, worth $1,24 billion in 2010, maintaining its position as the world's largest small kitchen appliance market, in both volume and value terms, for the next five years.

"90% of the rice cooker market currently lies in the Asia-Pacific region, and penetration rates are close to 100% in many of these countries." commented Ann Bird, the report author at IMS Research. Hence, the rice cooker market in Asia-Pacific is mainly driven by replacement. Growth opportunities exist in America and Europe, as ethnic cooking has become more and more popular. "There seems to be lots of potential here, which is perhaps not currently being fully exploited by the major OEMs." continued Bird.

This information is presented in IMS Research's recently published and highly detailed report The Worldwide Market for Small Household Appliances - 2006 Edition. This report looks individually at the global market for 22 small household appliances, as well as the technology used within these appliances.

http://www.marketresearchworld.net/index.php?option=com_content&task=view&id=916&Itemid=77

Neuro Fuzzy Logic Rice Cooker

- **"Fuzzy" works well even if control math is unknown**
- **"Neuro" (as in neural nets) adapts control as components age**
  - When neuro-fuzzy came out, everyone tossed their perfectly good fuzzy rice makers out to buy new ones!

http://www.makeuseof.com/tag/9-stupidest-smart-home-appliances/

Panasonic Handset-Enabled Rice Cooker

Plain, boiled rice is one of the easiest dishes you can cook.

God help you if you're so dense you need to spend over $1000 on a WiFi enabled rice cooker to make the culinary equivalent of playing chopsticks on the piano. But some people are, which explains the existence of the Panasonic Handset-Enabled Rice Cooker.

It be controlled through a FeliCa enabled smartphone (like NFC, but even fewer people use it), and can store up to 100 recipes by connecting to the Panasonic Cooking Cloud Server. No, I'm not making this up.

Preview

- **Feedback control**
  - Motivation
  - Bang-bang control
  - Hysteresis

- **PID control**
  - Proportional component
  - Integral component
  - Derivative component

- **Fuzzy logic**
  - Just the basics

Overview Of Control

- **How do you know what actuator position to command?**
  - .. to avoid burning the rice
  - .. to get the right temperature in a classroom
  - .. to avoid scalding children with a water heater
  - .. to get cruise control at the right speed
  - .. to get the right fuel/air mixture for engine combustion
  - .. to get good flight performance from a fly-by-wire aircraft
    - military fighter jets are dynamically unstable: they “want” to fall out of the sky(!)

- **Basic idea of control:**
  1. Read sensors
  2. Figure out the difference between what the sensors say and what you want
  3. Command actuators to change system
  4. Go back to step 1 (i.e., this is a control ***loop***)

Open Loop Control

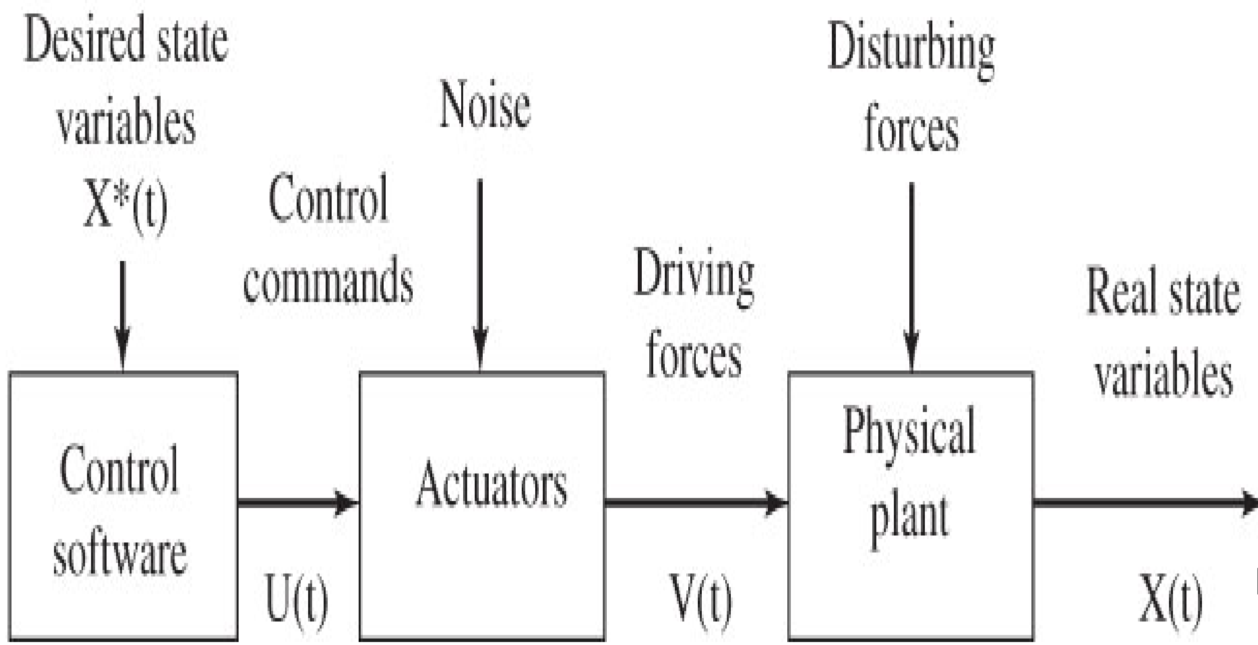


Figure 13.2
Block diagram of a microcomputer-based open-loop control system.

[Valvano]

- **No sensors – just actuators**
  - Problem: noise not taken into account
  - Problem: disturbing forces not taken into account
  - Problem: faults/failures in system not detected
  - Really have no way to know what is really happening is what you'd like happen!
  - Example: stepper motor (what happens if stepper motor slips a notch?)
    - One solution – have a mechanical "end stop" and make sure you run into it to reset

Skipping Stepper Moter Steps On a 3-D Printer

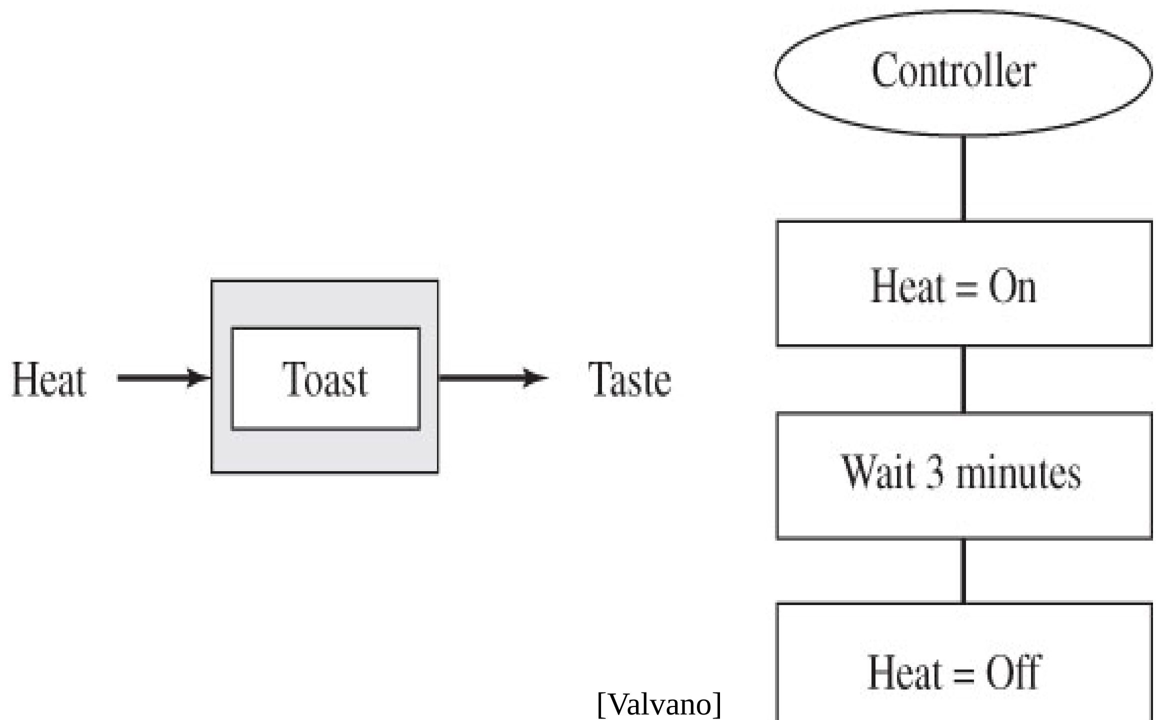
- **3-D printers use open-loop control on nozzle placement**

http://www.soliforum.com/topic/84/missing-steps/

Open Loop Control Applications

- **Open loop control OK for low-precision "simple" systems**
  - Classical toaster darkness knob is still open loop – it's just a timer
    - Toaster doesn't know how toast actually turns out!
  - Microwave with person setting time is open loop
    - Microwave with steam sensor is closed loop (next slide)
  - Can work well with highly repeatable processes/systems
    - Few disturbances; only minor noise

Figure 13.3
A simple open-loop control system.

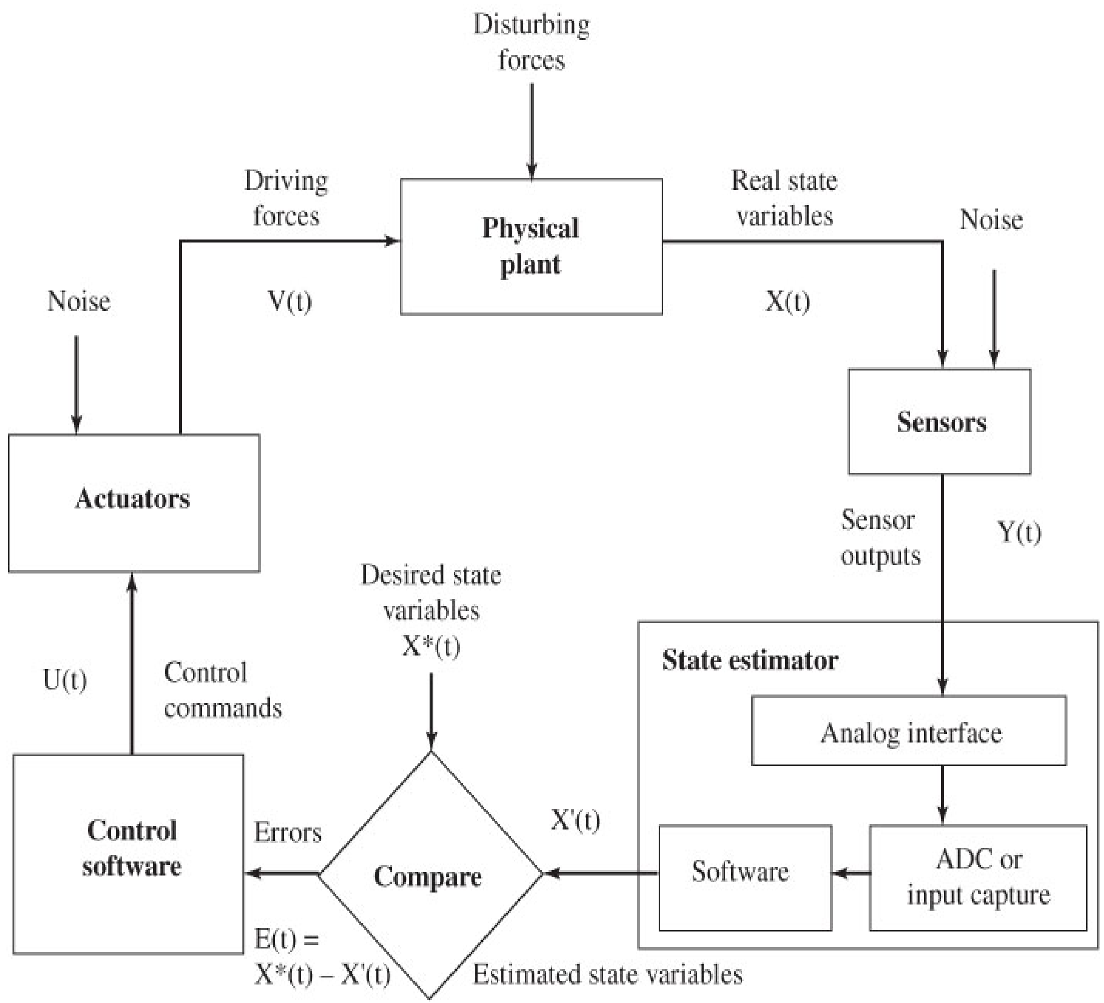


[Valvano]

Closed Loop Control

- **Idea: compare sensor values to desired values and change actuators**
  - "state" = the physical state of the system being controlled
  - System creates a control command to minimize error E(t)

Figure 13.1
Block diagram of a microcomputer-based closed-loop control system.



Some Big Control Issues

- **Imperfect state information** (how do you do state estimation?)
  - Sensors can’t tell you every possible thing
  - Sometimes there are sensors you wish you had, but can’t afford
  - Sometimes system is non-homogeneous, but you only have one sensor
  - We’re just going to worry about one sensor and assume it is perfect

- **Stability**
  - If control system is slower than state changes, you can destabilize the system
    - Keeping small “control lag” is essential for stability
  - We’re going to assume that you compute faster than needed

- **How do you compute control commands?**
  - Need a mathematical model of system response (“system identification”)
  - Somehow you know the state “setpoints” – we’re assuming single number
  - Compute difference between estimated state and desired state
  - Somehow you know how system is expected to respond
  - Issue a control command – focus on this in next several slides

Bang-Bang Control

- **Simplest closed-loop control**
  - Assume a sensor that directly senses desired state variable
  - Assume an actuator that can either increase or decrease state variable

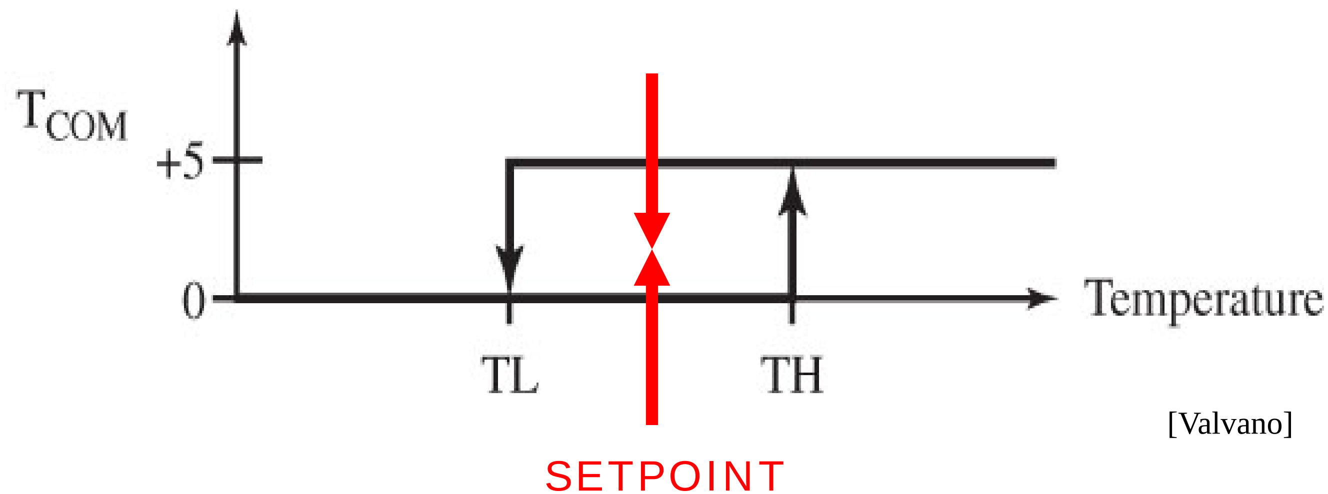
- **Control strategy:**
  - If sensed state variable < desired value ➔ actuator set to “increase”
  - If sensed state variable > desired value ➔ actuator set to “decrease”
  - (don’t worry about the “exactly ==” case for now)

- **As with other closed loop control – run this periodically**
  - Speed depends on time constants of system
  - If affordable, run control loops 10x faster than system can respond
    - If slower than that, you have to think harder about getting it right
    - (More technically: 10x faster than step response time)

What About "Exactly" Equal?

- **Analog values have noise**
  - When state is nearly equal to set point, will get fluctuations
  - Don't really want bang-bang controller rattling around the set point
  - Idea – put in a "dead band" around set point where no changes are made
    - Better term is saying the system has "hysteresis"

Figure 3.24
Thermostat control response with hysteresis.

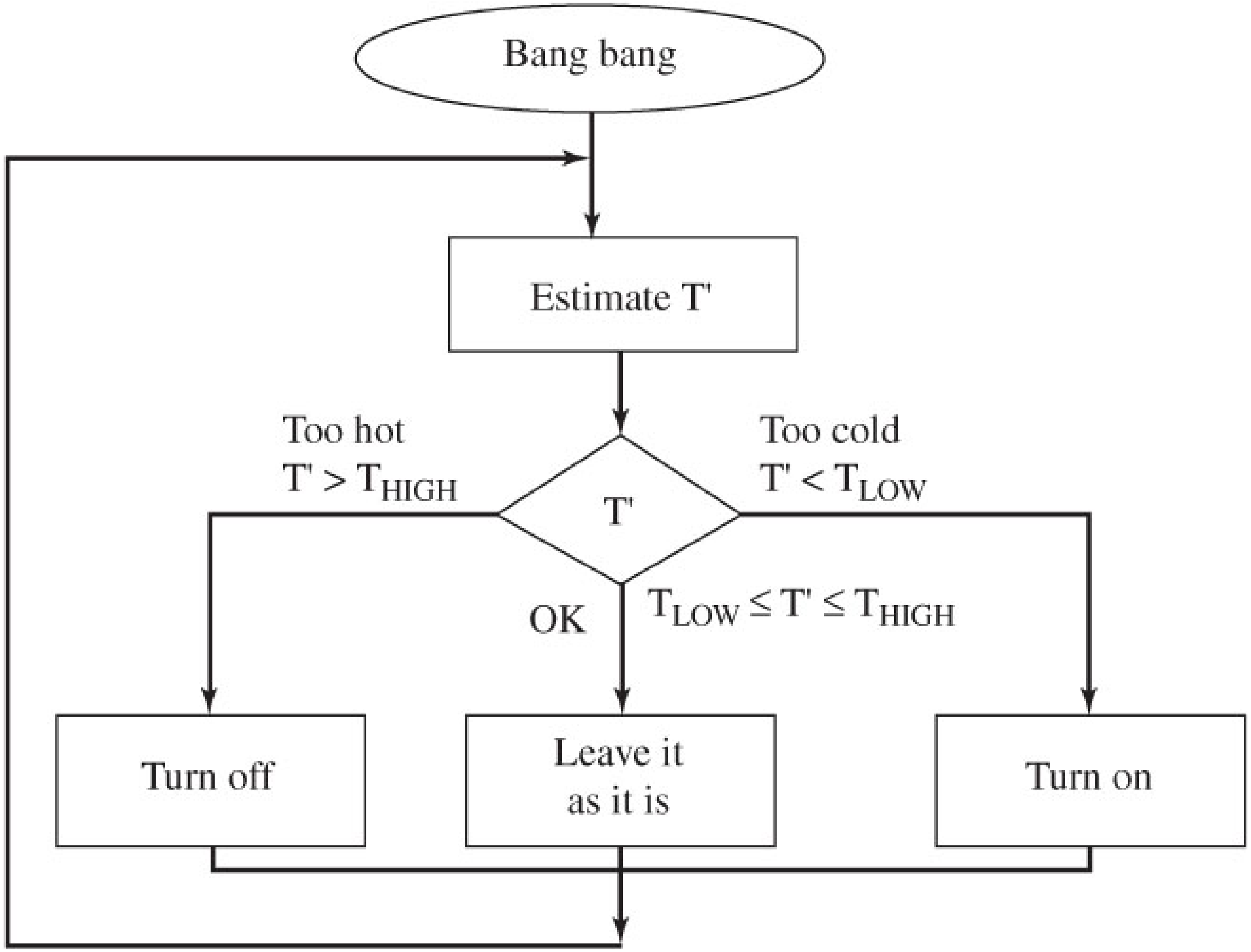


[Valvano]

- Example, winter thermostat operation
  - Heat room to 1 degree above setpoint, then turn off heat
  - Let room cool to 1 degree below setpoint, then turn on heat
  - Hysteresis = 2 degrees   (1 degree above to 1 degree below)

Bang-Bang Control With Hysteresis

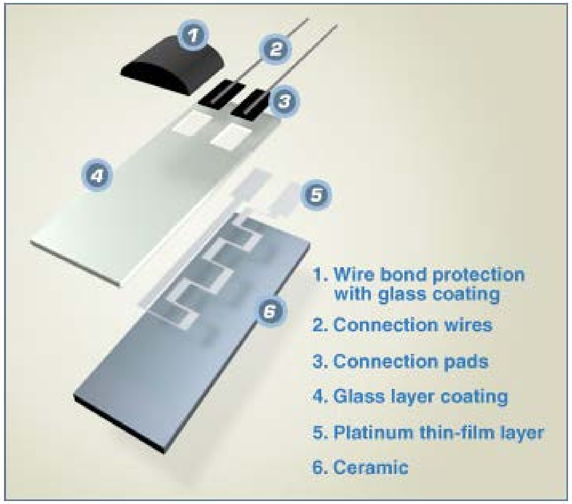
Figure 13.7
Flowchart of a bang-bang temperature controller.



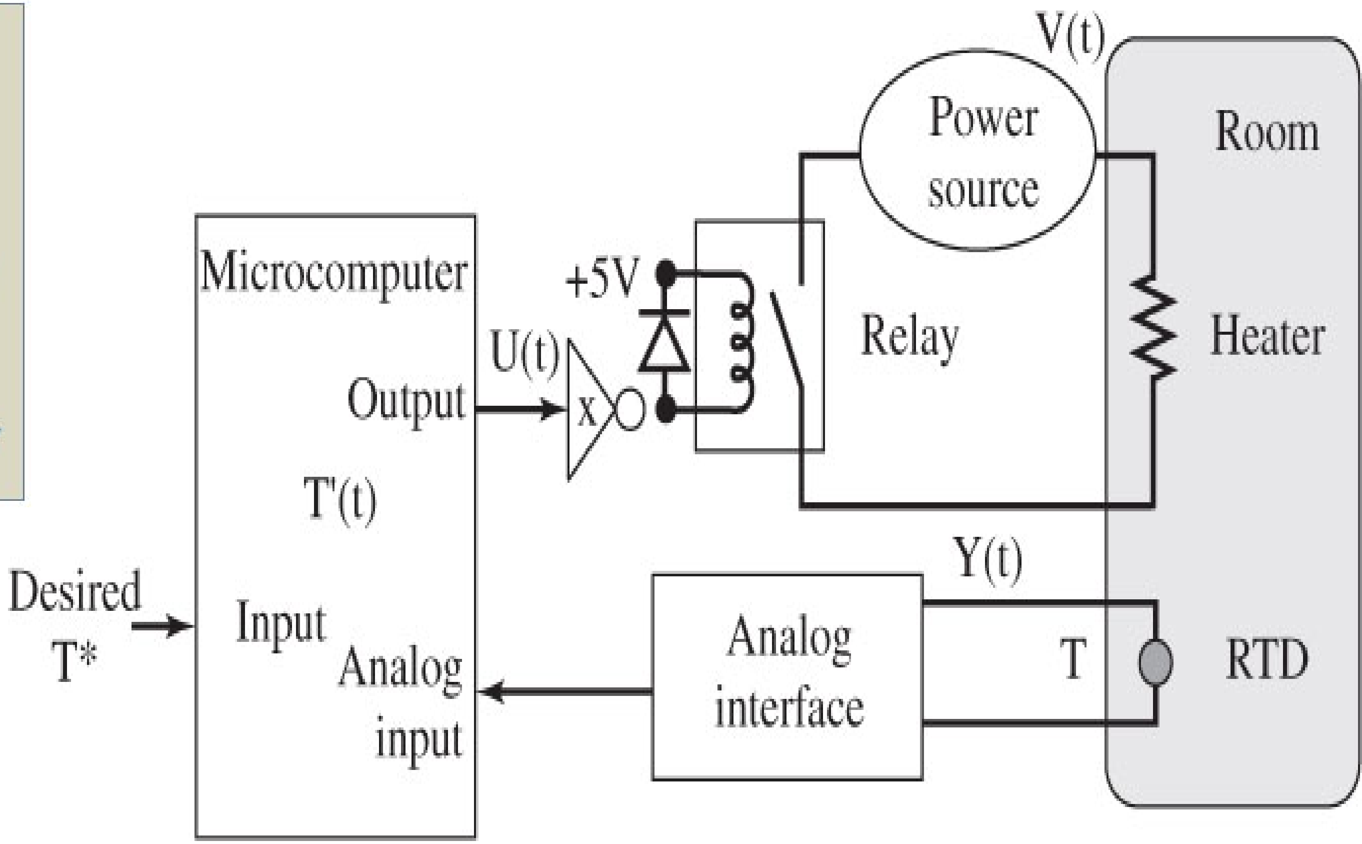
(would this be simpler as a state chart?)

Example Room Heater

- **RTD = Resistance Temperature Detector**
  - Resistance proportional to temperature
  - Older: very thin coiled wire in a protective case
  - Newer: very small amount of platinum

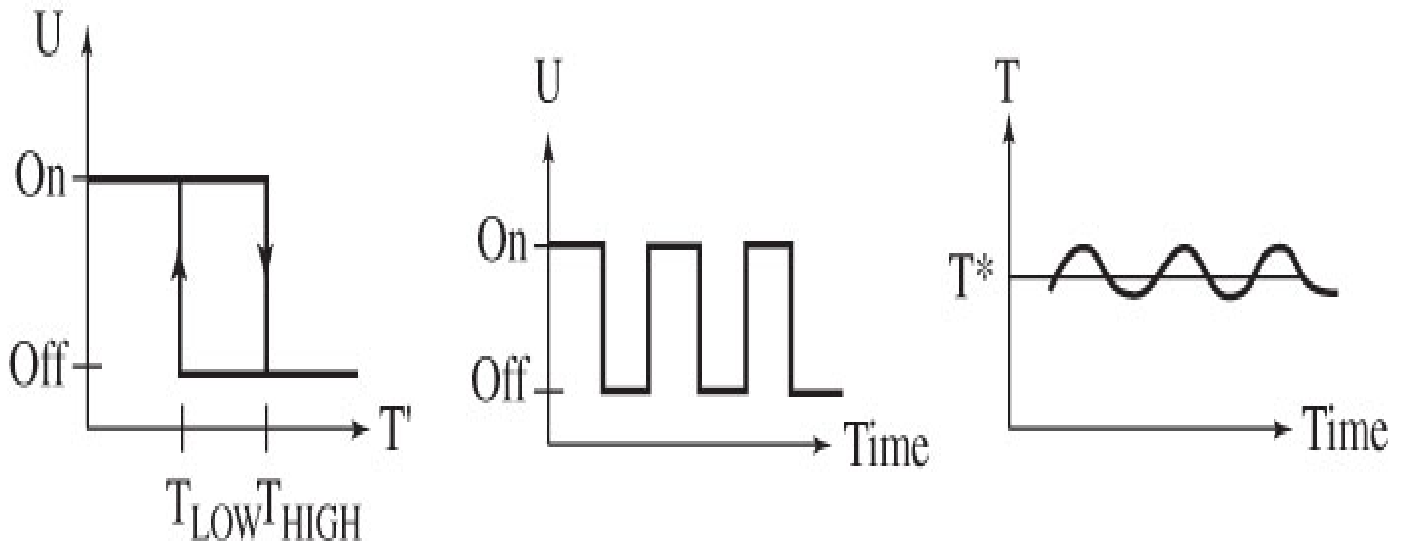


[Omega.com]



Bang Bang Tradeoffs

Figure 13.8
Algorithm and response of bang-bang temperature controller.

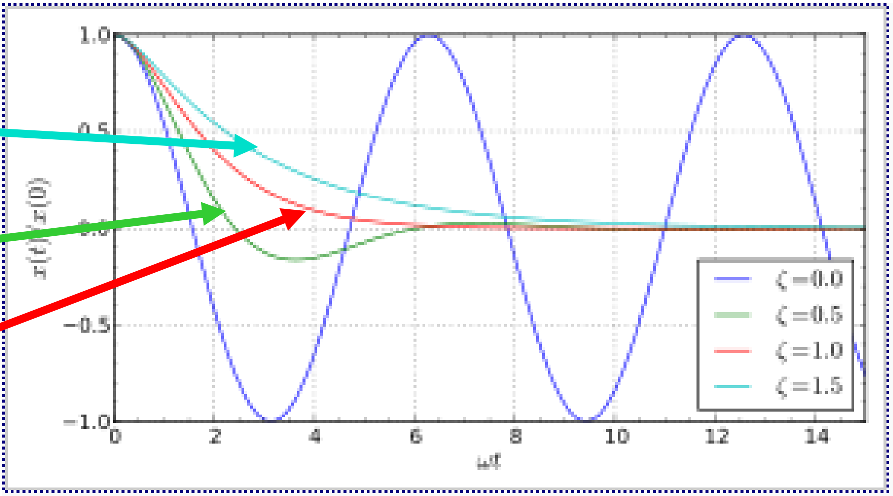


- **Advantages**
  - Simple
  - Effective in many situations – usually involving a relatively constant set-point

- **Disadvantages**
  - Always uses maximum control energy – might be wasteful
  - Assumes system naturally acts as a low pass filter – introduces "vibration"
    - Assumes actuator force is small compared to system inertia at the sample rate

Damping

- **Damping is the attenuation of oscillations in system response**
  - For control, has to do with how system oscillates as it approaches set point



Time dependence of the system behavior on the value of the damping ratio ζ, for undamped *(blue)*, under-damped *(green)*, critically damped *(red)*, and over-damped *(cyan)* cases, for zero-velocity initial condition.

- **Over-damped**
  - System takes a long time to reach set point
- **Under-damped**
  - System overshoots setpoint
- **Critically damped**
  - System reaches set point quickly and doesn’t overshoot
  - (Use math from control systems to figure out the right damping ratio)
- **How can we do better than bang-bang controller?**
  - What kind of damping would you expect a bang-bang controller to have?
  - What could be done better?

Proportional Control

- **Idea: amount of actuation proportional to amount of feedback error**
  - If you are way off set-point, use full actuator force
  - If close to set-point, use just a little actuator force
- **Example: air conditioner**
  - Usually chilled water in system is kept at constant temperature (perhaps 50°F)
  - Vary fan speed in heat exchange in proportion to how far off air temp is
    - Hot room gives hot heat exchange input air temperature; cool room doesn't
    - Might want to get constant exhaust air temperature (e.g., for cooling computers)
    - Thermal energy transfer rate is controlled by fan speed, not water temperature
    - Control exhaust air temperature by varying fan speed (speed up fan if exhaust is too hot)
- **Important notion for control systems – parameter tuning**
  - How far away from set-point corresponds to 100% actuator change force?
  - How close to set-point is 0% actuator change force? ("dead band")
- **Requires proportional actuator**
  - Can you use this approach with a purelydigital output? (yes – how?)
  - Can we do even better? – Yes; stay tuned…

Control Variables

Y(t)= Sensor values

X*(t)= Desired state of system

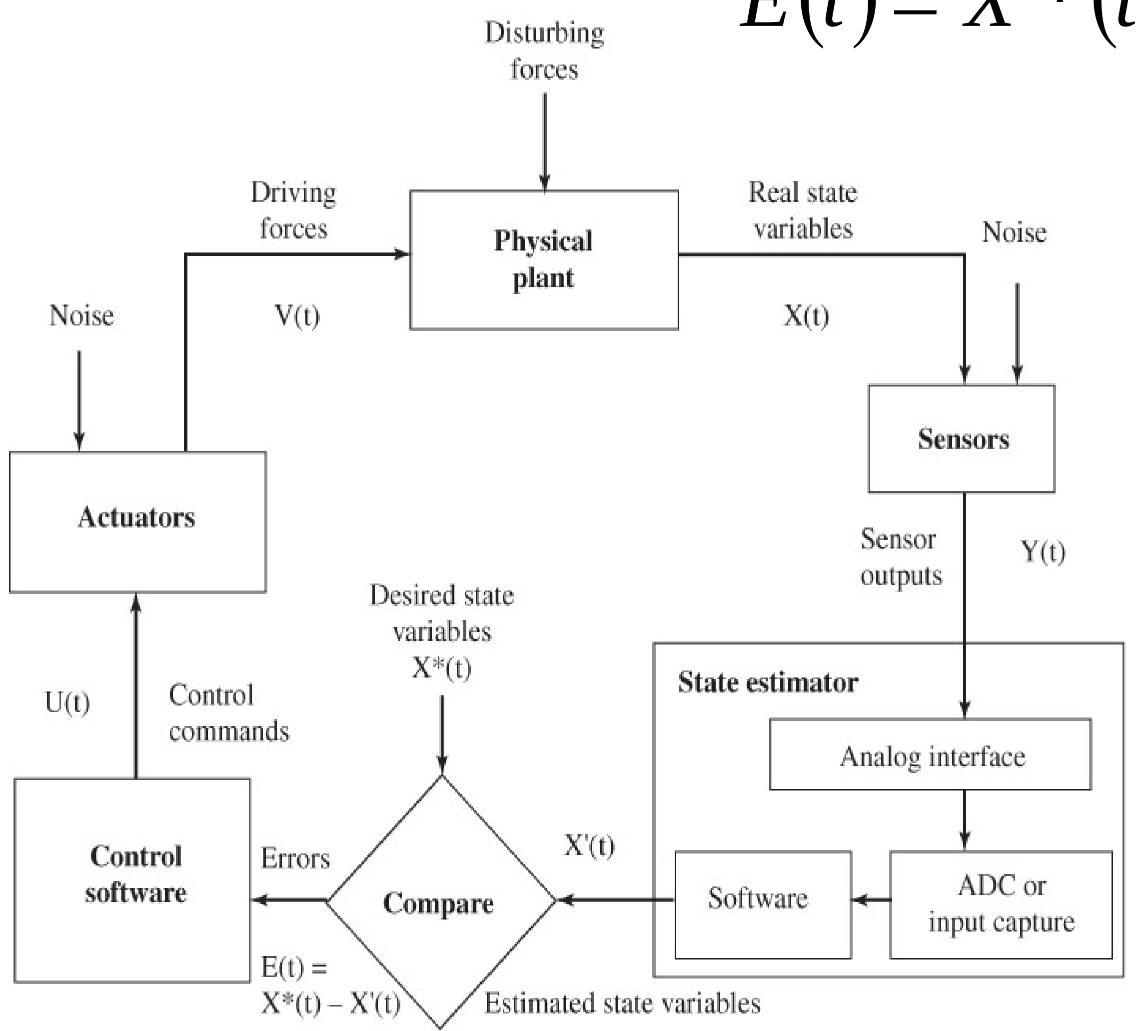
X'(t)= Estimated state of system

U(t)= Actuator Commands

E(t)= Error

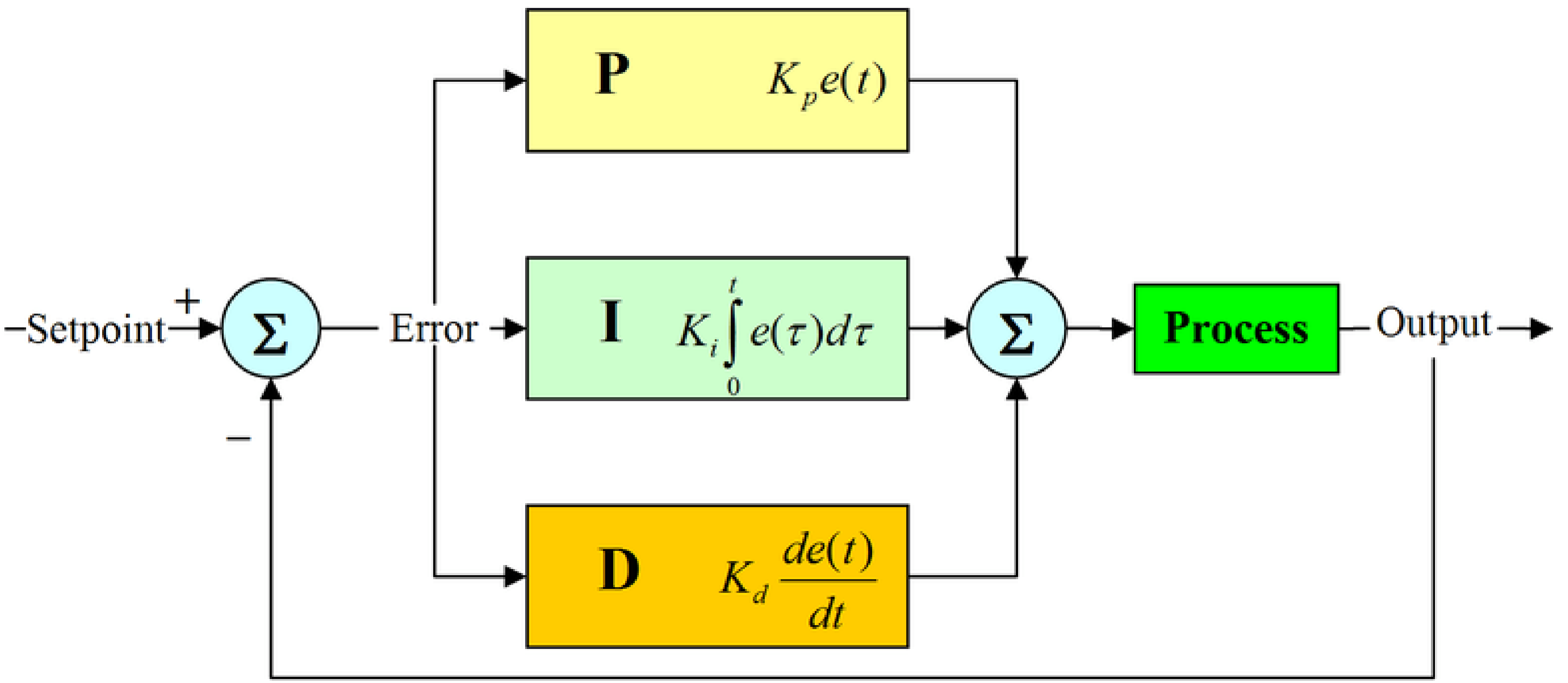
$$E(t) = X^*(t) - X'(t)$$

Figure 13.1
Block diagram of a microcomputer-based closed-loop control system.



Building Up To A PID Controller

- **PID ➔ Proportional + Integral + Derivative**



[Wikipedia]

Proportional Control

- **Recall: proportional control response U(t) is proportional to error. Mathematically:**

$$U(t) = K_p E(t)$$

- **For this terminology U(t) refers to actuator that:**
  - Changes system in direction of U(t) (e.g., + ➔ heat; – ➔ cool)
  - Has proportional ability (multiple levels, or PWM output)
  - So, this variable is rate of energy being added to/subtracted from system

- **Value of K_p is chosen based on system dynamics**
  - In a real system, U(t) might saturate actuators (e.g., limited by BTUs)
  - Special case – bang-bang means control Kp is “infinite”
  - What if K_p is too small; too large?

Adding An Integrator

- **Proportional system can be twitchy**
  - Noise in system will cause variations that might take care of themselves
  - Straight proportional control will react to every little noise event
  - Solution – add an Integrator to smooth things out
  - In effect, add a low pass filter to the control equation

- **Proportional system might not apply enough force**
  - A disturbance might be stronger than the proportional force applied
  - Want to keep history –
the longer you're away from setpoint, the harder system pushes

- **K_I ➔ integrator tuning constant**
  - What if K_I is too small? Too large?
  - How big is the integrator window?

$$U(t) = K_p E(t) + K_I \int_{t-\delta}^{t} E(\tau) d\tau$$

Adding A Derivative Component

- **A good hockey player plays where the puck is. A great hockey player plays where the puck is going to be. – Wayne Gretzky**

- **Proportional + Integrator system might not be enough**
  - You want to aim where the target is going to be, not where it is

- **Add a derivative term**
  - "Anticipates" future error – based on whether error is increasing or decreasing
  - Have to be careful to avoid over-doing this term, else makes system twitchy

- **K_D ➔ derivative tuning constant**
  - What if K_D is too small? Too large?

- **Result is a PID controller (Proportional + Integral + Derivative)**

$$U(t) = K_p E(t) + K_I \int_0^t E(\tau)\, d\tau + K_D \frac{dE(t)}{dt}$$

PID Controller Example

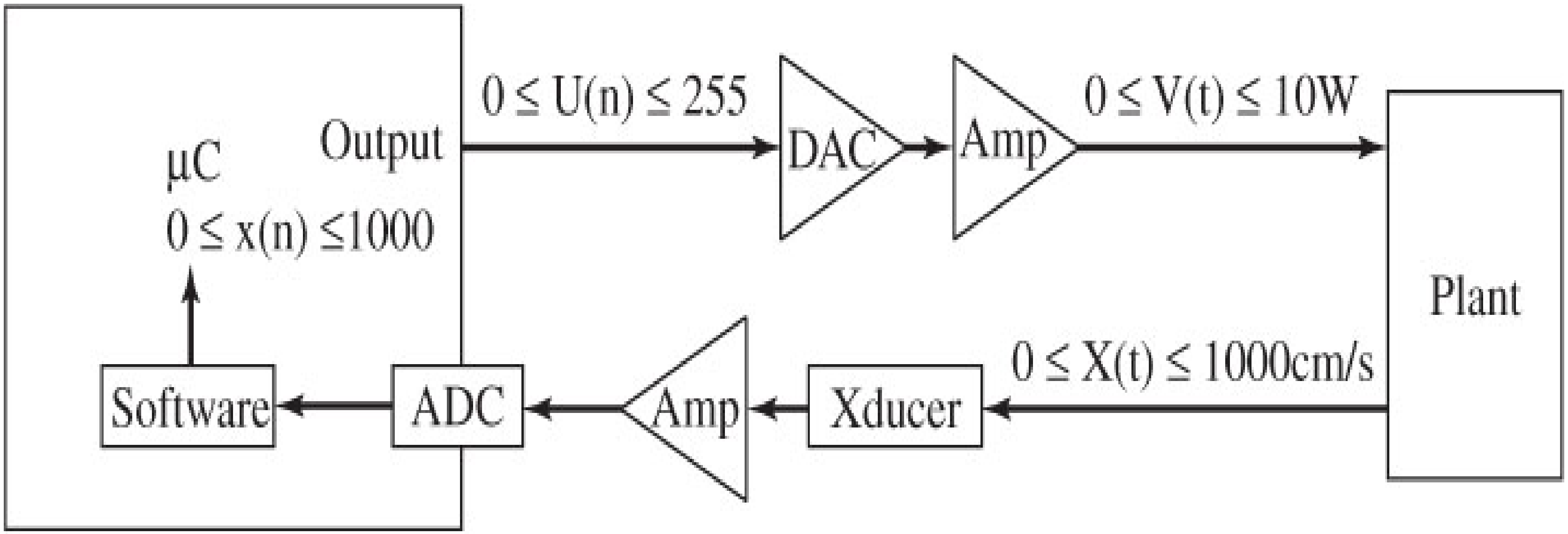


Figure 13.15
Interface of a PID velocity controller.

- **How do you set the constants?**
  - Plan A: guess and fiddle around with it
  - Plan B: read a random web article on it, then fiddle around with it
  - Plan C: take a controls course (and become an expert at fiddling around with it)

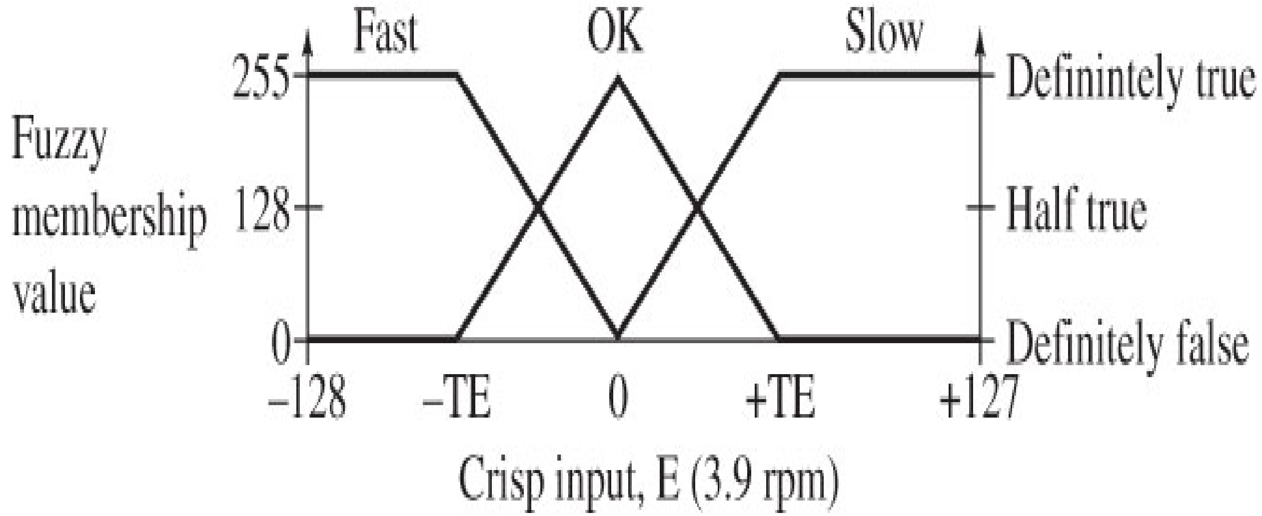
Fuzzy Logic: Control For The Math-Impaired

- **Sometimes you want a "good enough" technique**
  - If you aren't a control math whiz
  - If you don't have enough information ("system ID") to plug into math
  - There are a lot of inputs available, and control math is just to hairy to do
  - A gross oversimplification: this is proportional control with big quanta
    - In all fairness, its strength is simplifying *multi-sensor* control

- **What's the "fuzzy" part?**
  - Inputs and outputs have "fuzzy" values instead of "crisp" values
    - Example crisp values: temp=65F ;   voltage=17V;   speed=53mph
    - Example fuzzy values: temp=very hot; voltage=low;   speed=medium
  - AND might have several values to some degree
    - Example, if temp = {cold, cool, OK, warm, hot}   these are valid fuzzy values:
      - » Cold+cool   (somewhere between cold and cool)
      - » OK+warm   (just a little bit warm)
      - » warm+hot   (pretty warm, but not quite hot)

Step 1: Fuzzification Of Sensor Values

Figure 13.19
Fuzzification of the error input.

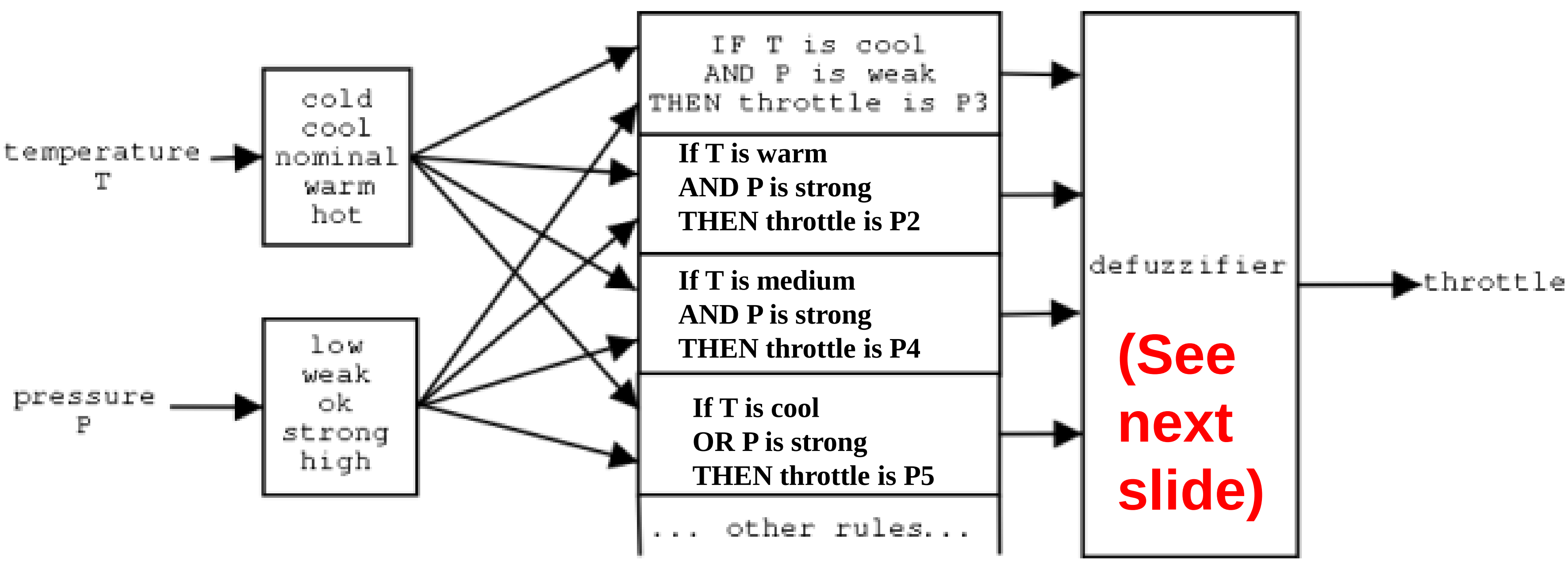


- **Sensor value is mapped into *<u>one or more</u>* bins (e.g., Fast, OK, Slow)**
  - -128 is "fast"
  - -5 is "a little fast, but also mostly OK" (if –TE is, say, -50)

- **The degree of membership per bin is given by the fuzzification process**
  - For each input, degree of membership is height of curve for that bin
    - So, you might be 25% "fast" and 75% "OK" at the same time ➔ total is 100%
    - In example below, you are only 100% "OK" at one point in center

Step 2: Fuzzy Control Rules

- **Set of rules used to determine output**
  - Each rule "fires" in proportion to how true its inputs are
    - "AND" is minimum of inputs; "OR" is maximum of inputs
    - All rules fire every time, but many rules weighted zero if "IF" side results in 0
  - Examples using below figure:
    - T=25% "cool" AND P=75% "weak" would give 25% of P3 (AND is min)
    - T=25% "cool" OR P=75% "strong" would give 75% of P5 (OR is max)



http://en.wikipedia.org/wiki/Fuzzy_control_system ; edited to eliminate repeated rules

Step 3: Defuzzification To Produce Outputs

- **Defuzzification produces the actuator output value**
  - Some sort of weighted average of all the rules that are triggered
  - This part is more or less magic – pick one you like
    - There are many to choose from

- **For example, if rule output is:**

10% P2 ; 25% P3; 40% P4; 25% P5

(10% very slow; 25% slow; 40% medium; 25% fast)

and output values map to:

very slow = 10; slow = 20; medium = 35; fast = 50

a simple rule is:

10%*10 + 25%*20 + 40%*35 + 25%*50 → output of 32.5

Chapter 9 of HC12 instruction set manual explains how Freescale does things

REVW Fuzzy Logic Rule Evaluation (Weighted) REVW

Operation:

MIN-MAX Rule Evaluation with Optional Rule Weighting

Description:

REVW performs either weighted or unweighted evaluation of a list of rules, using fuzzy inputs to produce fuzzy outputs. REVW can be interrupted, so it does not adversely affect interrupt latency.

For REVW to execute correctly, each rule in the knowledge base must consist of a table of 16-bit antecedent pointers followed by a table of 16-bit consequent pointers. The value $FFFE marks boundaries between antecedents and consequents, and between successive rules. The value $FFFF marks the end of the rule list. REVW can evaluate any number of rules with any number of inputs and outputs.

… two pages of stuff …

Source Form	Address Mode	Object Code	Access Detail(1) HCS12	Access Detail(1) M68HC12

REVW
(add 2 at end of ins if wts)
(replace comma if interrupted)
	Special	`18 3B`	`ORf(t,Tx)O`
			`(r,RfRf)`
			`ffff + ORf(t,`
				`ORf(tTx)O`
				`(r,RfRf)`
				`ffff + ORf(t,`

1. The 3-cycle loop in parentheses expands to five cycles for separators when weighting is enabled. The loop is executed once for each element in the rule list. When an interrupt occurs, there is a 2-cycle exit sequence, a 4-cycle re-entry sequence, then execution resumes with a prefetch of the last antecedent or consequent being processed at the time of the interrupt.

Review

- **Feedback control**
  - How Bang-Bang control works
  - What's Hysteresis?

- **PID control**
  - Proportional component
  - Integral component
  - Derivative component
  - Equation for PID control and what each K term does

- **Fuzzy logic**
  - Based on levels of "truth" for various system states and outputs
  - Fuzzification; fuzzy rule application; defuzzification