

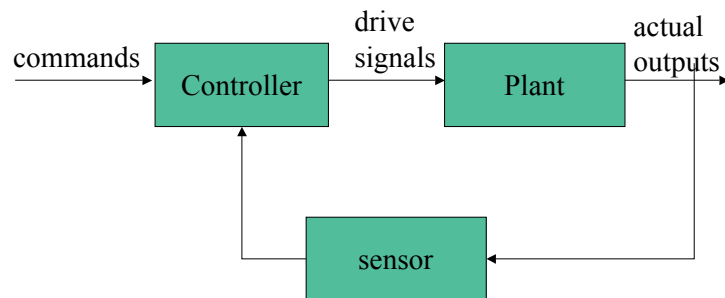
# Design and Analysis of Real Time Systems

Dr. Mantis Cheng  
Department of Computer Science  
University of Victoria  
(7 May 2002)

## Basic Feedback Control

- Open and Closed Loop Control
- Discrete and Continuous Model
- PID Control
- Finite State Control
- Hybrid Control
- Adaptive Control

## A Simplified Model



## Open and Closed Loop

- A closed loop control has *feedback* into the controller from the plant's outputs.
- In an open loop system, commands are *converted* simply into drive signals.
- In a closed loop system, actual outputs are *monitored continuously* in order to produce the *required* drive signals.

## Discrete and Continuous

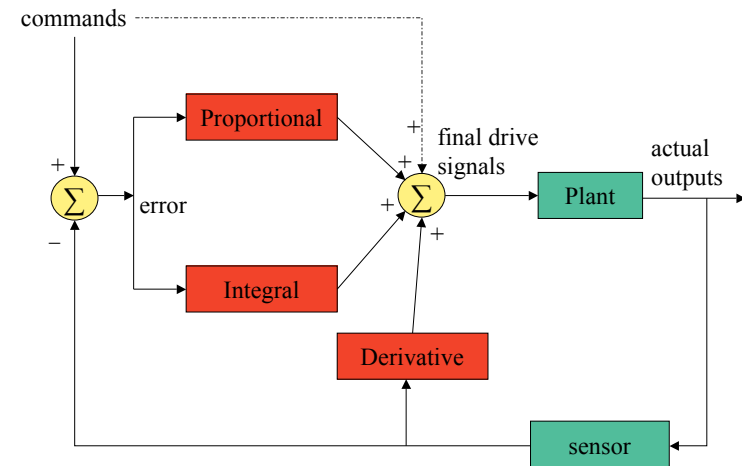
- **Continuous:** Stimuli (commands, outputs) and responses (drive signals) often maintain a continuous relationship (e.g., steering column and wheels).
- **Discrete:** A system may exhibit multiple modes of operations and discrete transitions between these modes (e.g., cruise control, washing machine cycles).

## PID Control

- PID stands for
  - **P**roportional (depends on *present* outputs)
  - **I**ntegral (depends on *past* outputs)
  - **D**erivative (depends on *future* output trends)
- Majority of simple *continuous* closed loop control problems can be solved by PID controllers.

## PID Controllers

- They are applicable to both *mechanical* as well as *electronics* devices.
- With inexpensive microcontrollers, A/D and D/A converters, *digital* PID controllers are increasingly more common.
- *Simple* PID controllers may be combined to form *complex* double, triple, ..., feedback control loops.



## PID Principles

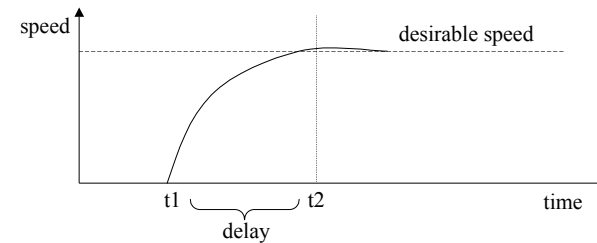
- A *command* gives rise to a *desirable* output.
- An (+/-) *error* is generated from comparing the *actual* against the *desirable* output.
- The errors are used by the P- and I- controllers to produce a P- and I- drive signals.
- Successive actual outputs are used by the D-controller to generate a D-drive signal.

M. Cheng

9

## A Simple DC Motor System

- Command: 60% of full speed (0..100 rpm)
- Drive signal: 3VDC (0.. 5VDC)
- Actual outputs:



M. Cheng

10

## Proportional Control

- The drive signal (p-drive) is *proportional* to the *current* error.

$$\text{error} = \text{desirable\_output} - \text{actual\_output}$$

$$\text{p-drive} = \text{error} * K_p$$

where  $K_p$  is a *constant gain*.

- For a given control problem, what is the *right* value of  $K_p$ ?

M. Cheng

11

## Proportional Gain

- With a *small*  $K_p$ , the actual output may converge to the desirable output *slowly*.
- But, with a *large*  $K_p$ , the actual output may *never* converge, i.e., the plant *oscillates*.
- Note that p-drive is *always* proportional to the error.
- For some problem, this technique may be inadequate to reach a desirable output.

M. Cheng

12

## Integral Control

- The drive signal (i-drive) is proportional to the *accumulated* errors.

$$ierror = ierror + error$$

$$i-drive = ierror * Ki$$

$$drive = i-drive + p-drive$$

where  $K_i$  is a *constant gain* with a range of (0.0001 .. 0.01).

- “ierror” may become *too* large; hence, it must be bounded.

## Integral Gain

- The accumulated errors depend on the sampling frequency.
- If  $K_i$  is *too* large, the i-drive signal may *saturate* the plant, i.e., drives it at the *extremes*.
- Integral control is seldom used by itself; it usually is combined with P-control.

## Derivative Control

- The P- and I-control examine the current and past errors.
- The Derivative control follows the *trend* of the outputs, i.e., the rate of change.
- By maintaining the *most recent* outputs at *fixed* intervals, we can *predict* the rate of change of the outputs.

## Derivative Gain

$$d-drive = K_d * (actual\_output - last\_output)$$

$$last\_output = actual\_output$$

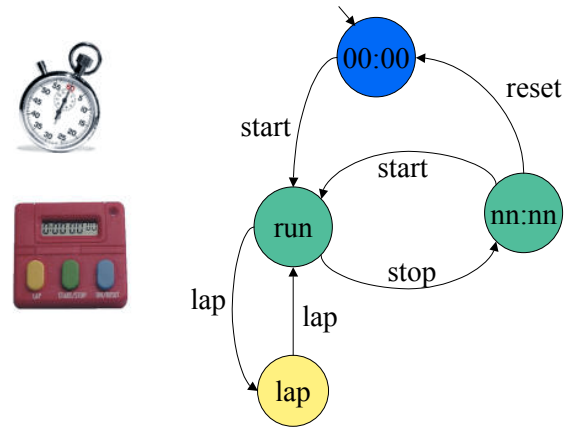
where  $K_d$  is a *constant gain* taking the sampling interval into account.

- The accuracy of the “*trend*” depends on the accuracy of the *sampled* time.
- D-control is useful when a plant has a *non-linear* rate of change of outputs.

## Finite State Control

- There are control problems where the commands/drive signals and the outputs are not *continuously* related. For example,
  - vending machines,
  - data/tele-communication systems,
  - security alarm systems,
  - traffic light controllers, etc.

## A Simple Stopwatch



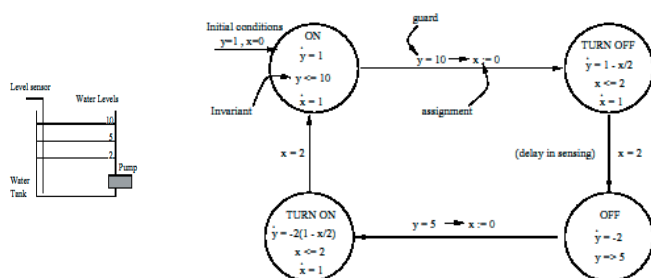
## Finite State Machines

- A FSM has a finite number of discrete (*abstract*) states, and a set of transitions which defines the “*allowed*” state changes.
- The set of transition labels defines its “*interface*” (alphabet).
- The sequences of allowed transitions define its *behaviors*.

## Hybrid Control

- A hybrid controller is a combination of finite state machines and differential equations.
- A “state” is a set of equations for a collection of continuous control variables.
- The switching from one set of equations to another is defined by an automaton.

## A Water Pump Problem



M. Cheng

21

## Adaptive Control

- Deterministic FSMs and Differential Equations are appropriate for *well-defined* control problems.
- What if:
  - we *do not* have a *good* or *complete* model of the “real” world;
  - the “real” world is very *noisy*, i.e., full of imprecise information;
  - we have *conflicting* decisions in our controllers.

M. Cheng

22

## Dealing with Noise

- Noise may be induced by the “*imperfect*” transducer’s interface to the real world.
- Noise may be reduced by using a suitable “*filter*”.
- Imprecise inputs are a form of “*noise*”.
- It is important to introduce “*safe guards*” for dealing with “*soft errors*”.

M. Cheng

23

## Fuzzy Control

- The controller is defined by a set of “expert” rules, not by equations.
- The control variables are *quantized* into *fuzzy sets*, e.g., heavy, medium, light. “*If traffic is light and it is not during rush hour, keep the green light on longer.*”
- Fuzzy sets allow imprecise inputs to be handled *discretely* with varying degrees.

M. Cheng

24

## Dealing with Conflicts

- How do we resolve the conflicts when multiple drive signals generated by different controllers at the same time?
  - Mixing;
  - Masking;
  - Preempting;
  - Inhibiting;
  - Merging, etc.

## Subsumption Architecture

- A subsumption operator is an “*arbitrator*” for resolving conflicting decisions.
- A *preferred ordering* (subsumption order) is defined on a set of behaviors/drive signals.
  - “*Whenever A and B occur simultaneously, always choose A and ignore B.*”
- Subsumption is a form of safe guard.