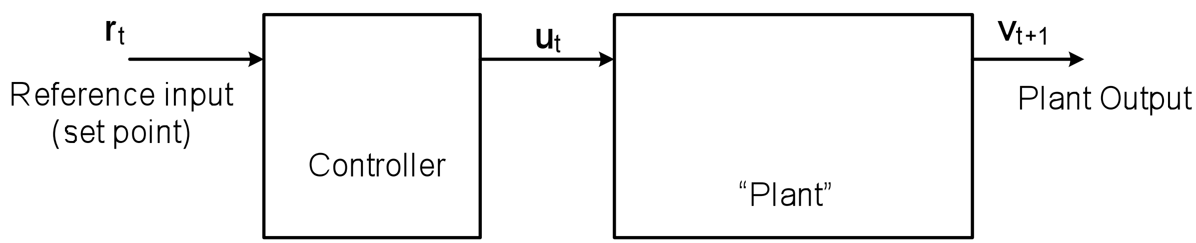
Lecture 9 – Implementing PID Controllers

CSE P567

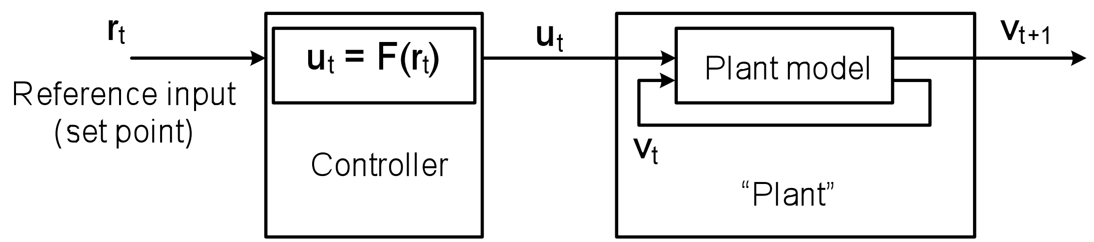
Control Systems

- We want to control some system
  - called a "Plant"
- There is an output value we want the system to achieve
- To be equal to a given goal value: set point
- Using a control input to the Plant



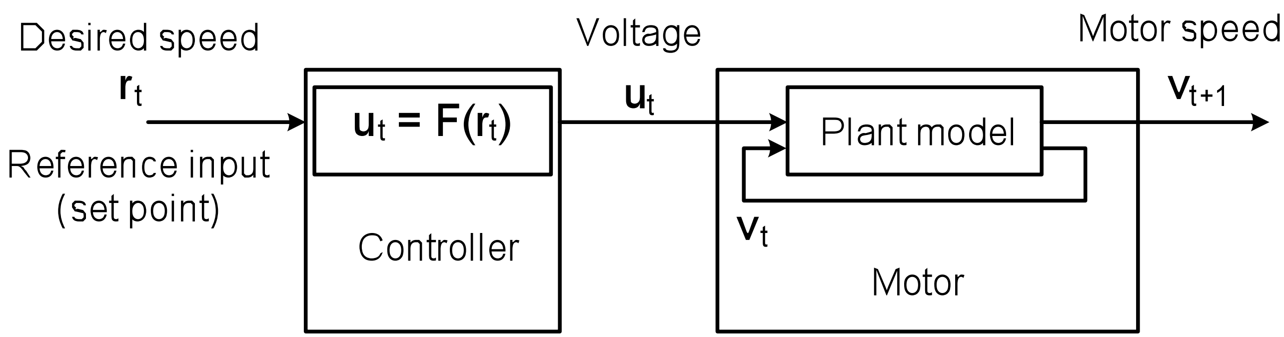
Control Systems

- If we have a good model of the Plant
- We can predict the output from the input
- Andthus we can compute the input needed to achieve a given output
  - $u_t = F(r_t)$
- In this case we can use an Open Loop Controller



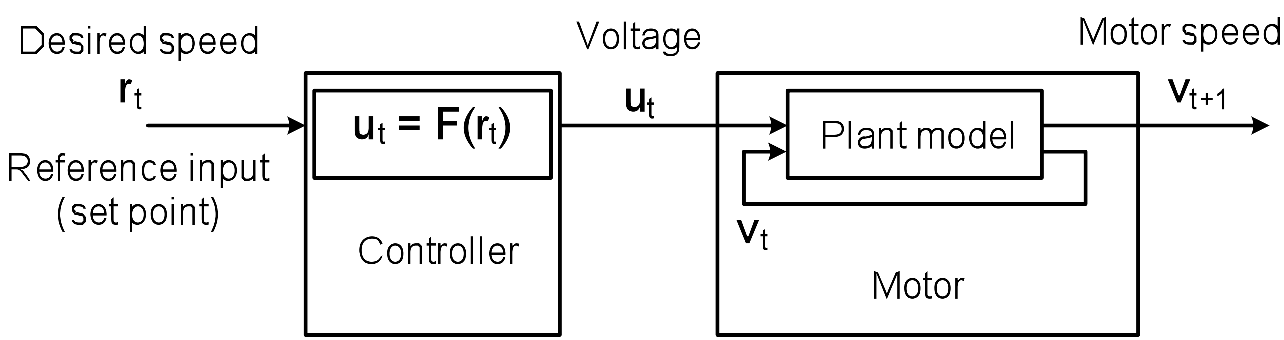
Example Open Loop System

- Motor example
- Motor model: $v_{t+1} = 0.7v_t + 0.5u_t + d_t$
  - d_t is any disturbance in the plant
- Control: $F(r_t) = P\, r_t$
  - Linear function



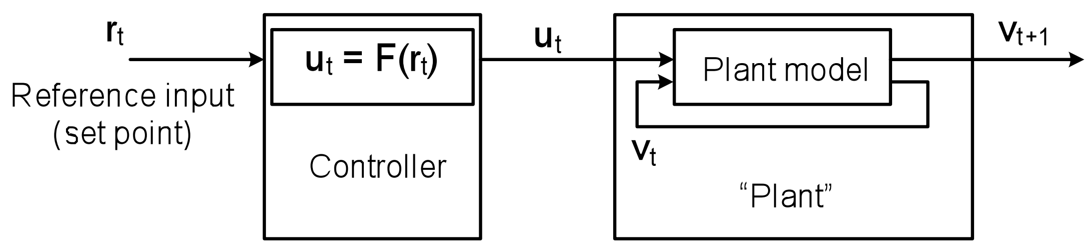
Example Open Loop System

- Control: $F(r_t) = P\, r_t$
- What should P be?
- In steady state:
  - $v_{ss} = 0.7 v_{ss} + 0.5P\, r_t$
  - $0.3 v_{ss} = 0.5P\, r_t$
  - $v_{ss} = 1.667P\, r_t$
  - $P = 0.6$ (steady state speed = set point)



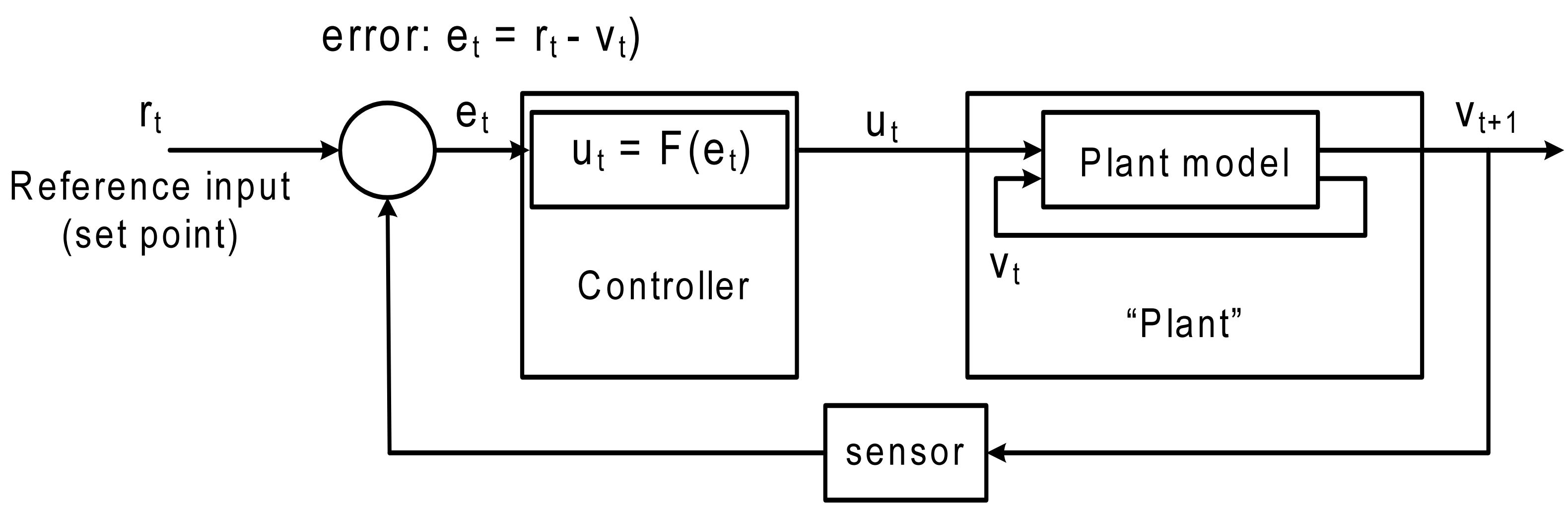
Problems with Open Loop Systems

- They fly "blind"
  - Dead reckoning
- Cannot respond to disturbances
  - Extra friction/load, wearout, etc.
- Cannot adjust to different Plants (motors)
  - Each has its own model
- Models may be difficult or impossible to derive



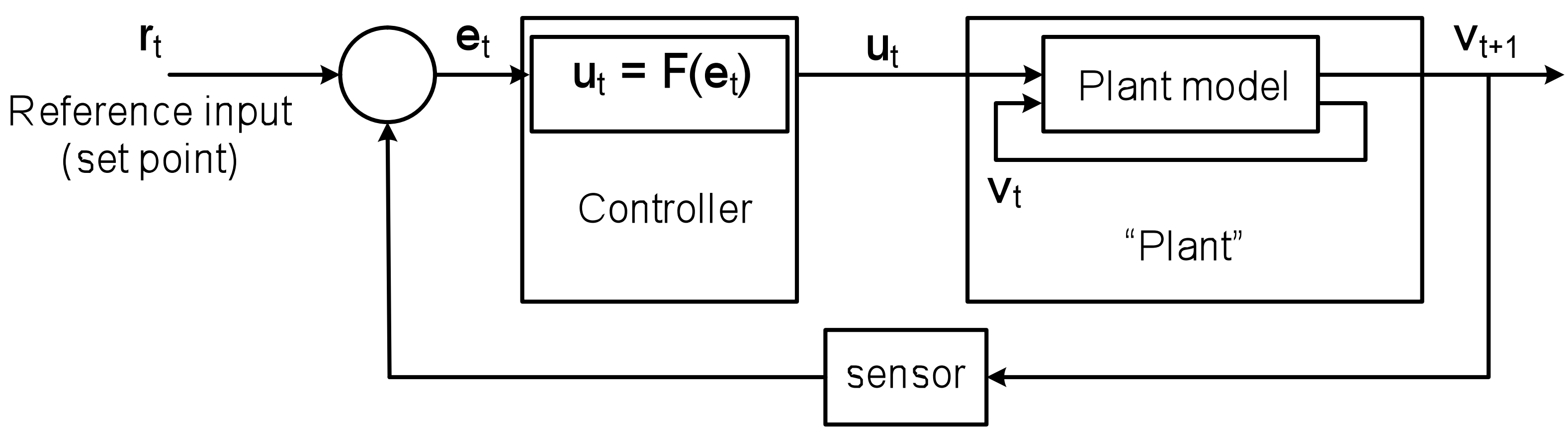
Closed Loop Systems

- Add feedback so controller knows results of actions
- We now know the difference between the set point and the output
  - And we try to make this zero



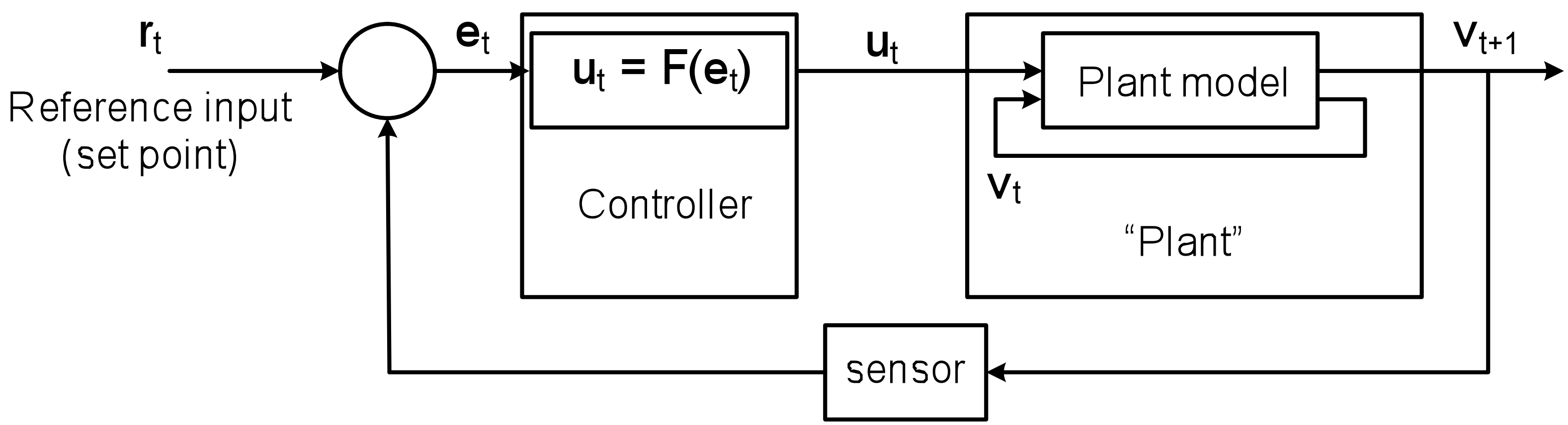
Proportional Controller

- Simplest controller
- $F(e_t) = K_p(e_t)$
- $v_{t+1} = 0.7v_t + 0.5\ K_p\ (r_t - v_t) + d_t$
- $v_{t+1} = (0.7 - 0.5\ K_p)\ v_t + 0.5\ K_p\ r_t + d_t$
- $\alpha = 0.7 - 0.5\ K_p$ determines whether v stays within bounds
- if $|\alpha| > 1$, then v_t grows without bound



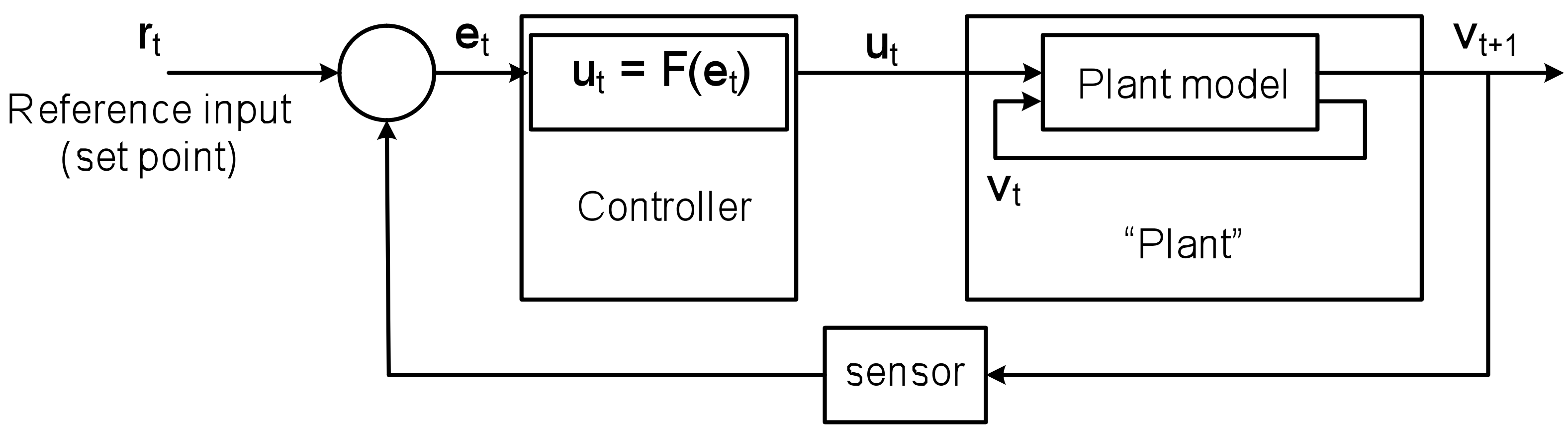
Proportional Controller

- $|\alpha = 0.7 - 0.5\,K_P| < 1$
- $K_P > -0.6$
- $K_P < 1.4$
- The best convergence rate is for Kp = 1.4
- But for Kp <= 1.4, we cannot reach the set point in steady state without oscillation



Adding a Integral Term

- How do we solve the problem where the output never reaches the goal?
- If we sum the total error over time, the output must reach the set point
  - with some overcompensation
- $F(e_t) = K_P(e_t) + K_I(e_0 + e_1 + e_2 + \quad + e_{t-1})$
- The Integral term allows the output to reach goal

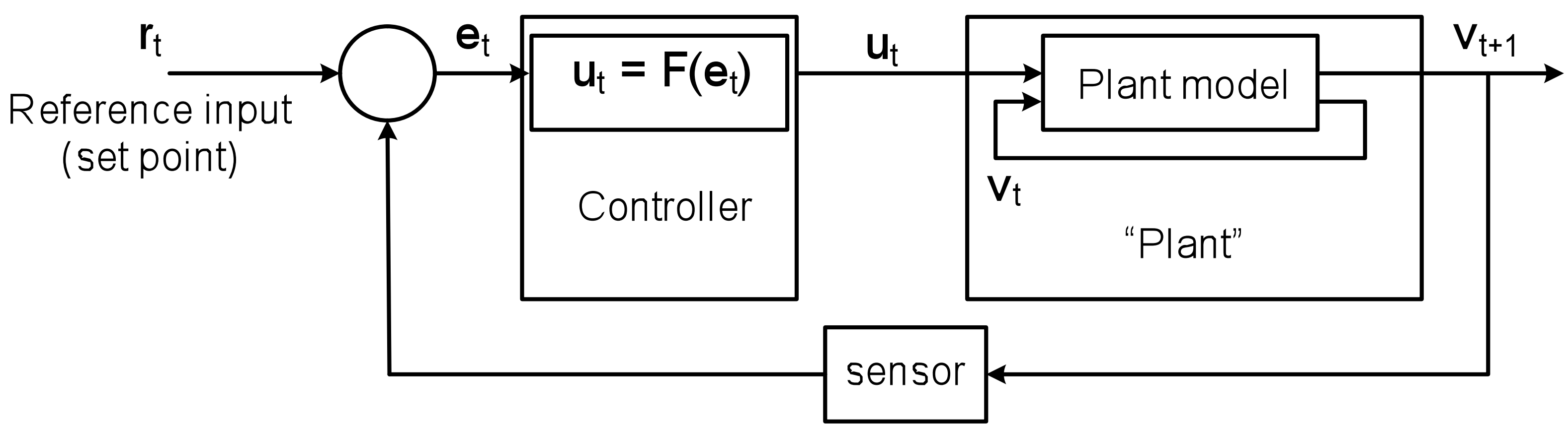


Integral "Windup"

- The integral term increases while the output is ramping up
- This causes overshoot
  - while the term "winds down"
- Can become oscillation
- Solution is to limit integral term
  - or cause it to die out
  - $I = 0.99I + e_t$

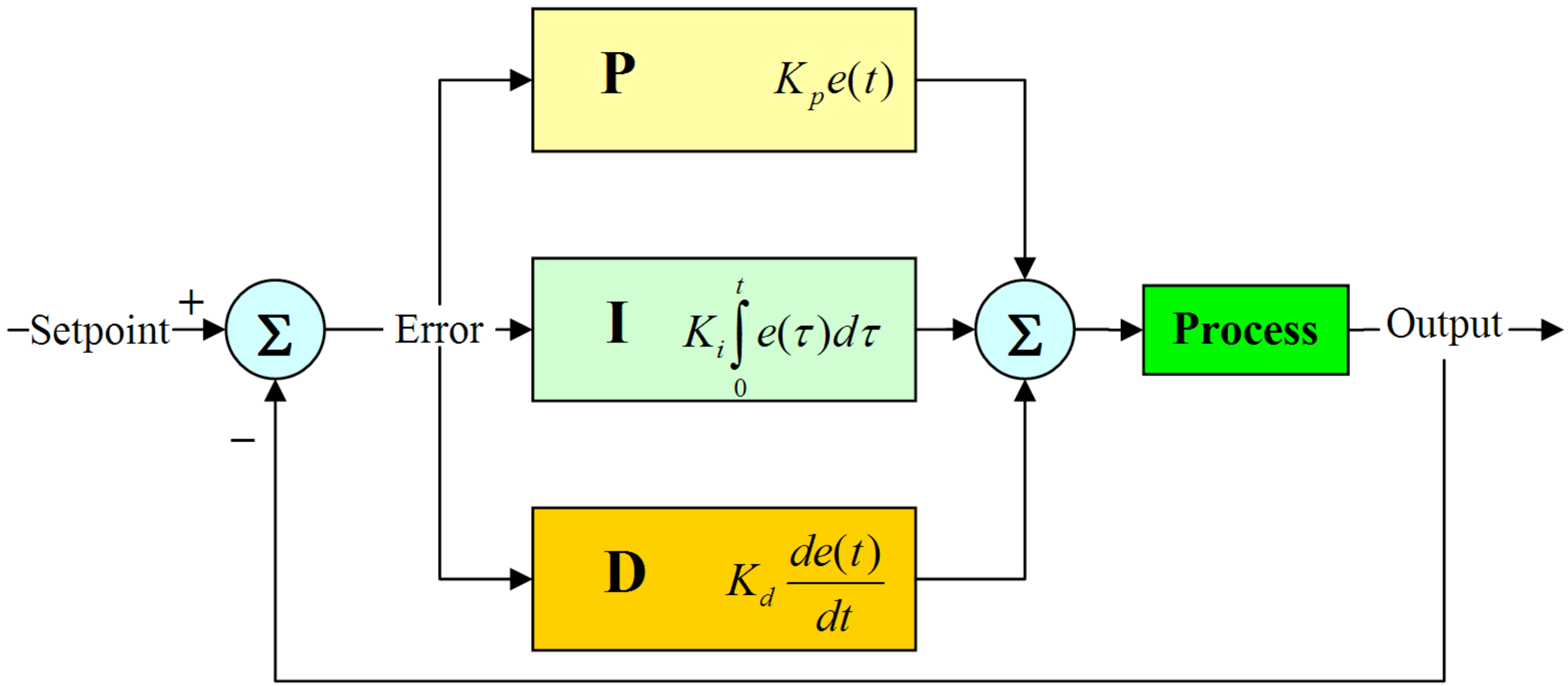
Adding a Derivative Term

- We look at rate of change of output:
  - If too slow, increase the control input
  - If too fast, decrease the control input
- $F(e_t) = K_P(e_t) + K_D(e_t - e_{t-1})$
- The Derivative term decreases oscillation
  - especially caused by disturbances



Summary of PID Controller

- We can build a PID controller that works well in practice in most situations without knowing control theory



Controller Performance

- Stability: The error variable should converge to a small number, preferably 0
  - i.e. little oscillation even with disturbances
- Performance:
  - Rise time/Response: e.g. 10% to 90% of final value
  - Peak time: Time to reach first peak
  - Overshoot: Amount in excess of final value
  - Settling time: Time before output settles to 1% of final value
- Disturbance rejection
- Robustness: Stability and performance should not be greatly compromised by small differences in plant or operating conditions

Tuning a PID Controller

- A search in 3 dimensions over all conditions
- If possible, use a large step function in the set point
  - e.g. 0 – 100%

- Heuristic procedure #1:
  - Set K_P to small value, K_D and K_I to 0
  - Increase K_D until oscillation, then decrease by factor of 2-4
  - Increase K_P until oscillation or overshoot, decrease by factor of 2-4
  - Increase K_I until oscillation or overshoot
  - Iterate

Tuning a PID Controller

- Heuristic procedure #2:
  - Set K_D and K_I to 0
  - Increase K_P until oscillation, then decrease by factor of 2-4
  - Increase K_I until loss of stability, then back off
  - Increase K_D to increase performance in response to disturbance
  - Iterate

Tuning a PID Controller

- Heuristic procedure #3:
  - Set K_D and K_I to 0
  - Increase K_P until oscillation:
    - $K_C = K_P$, P_C = period of oscillation
  - Set $K_P = 0.5\ K_C$
  - Set $K_D = K_P\ P_C / 8$
  - Set $K_I = 2K_P / P_C$

Implementing a PID Controller

- Can be done with analog components
- Microcontroller is much more flexible
- Pick a good sampling time: 1/10 to 1/100 of settling time
  - Should be relatively precise, within 1% – use a timer interrupt
  - Not too fast – variance in delta t
  - Not too slow – too much lag time
  - Sampling time changes relative effect of P, I and D
- Use interactive commands to set K_P, K_I, K_D
- Possible to program an adaptive PID controller
  - Perform a careful, automatic search
  - Must be able to control Plant offline
  - Complex