DC MOTOR CONTROL SYSTEMS FOR ROBOT APPLICATIONS

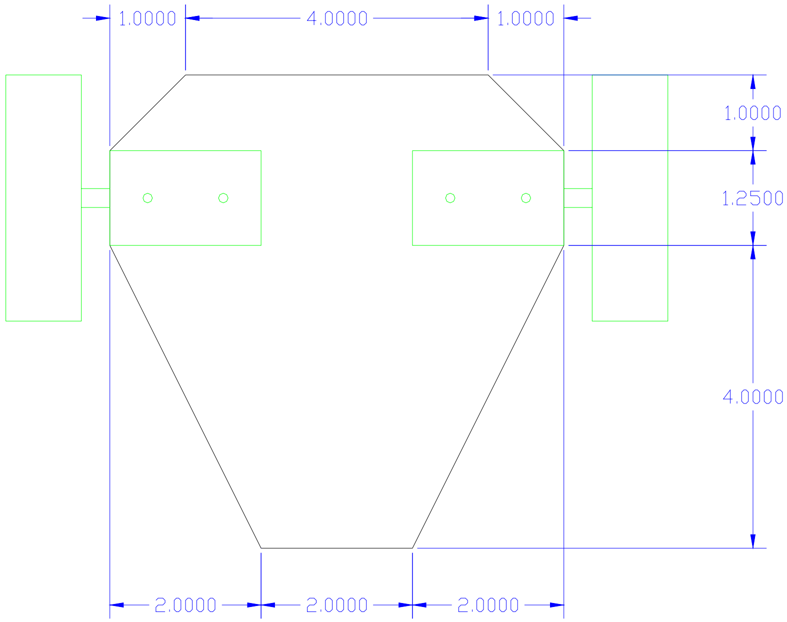
By: Rick Bickle

11/7/2003

Motor control questions

- Why do we need speed control?
- How is DC motor speed controlled?
- How is motor direction controlled?
- What circuits can be used?

Reasons for accurate speed control

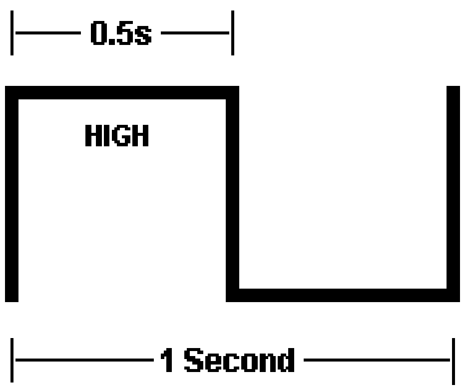


- Motor speed should be independent of load.
- Differential drive platforms need to synchronize wheel speed to go in a straight line.

Speed control with PWM

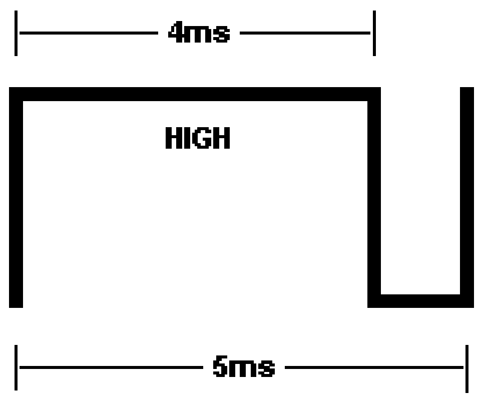
- Pulse Width Modulation

$$\text{Duty Cycle} = \frac{\text{Time Signal in High State}}{\text{Period of Cycle}} \times 100$$



$$\frac{0.5s}{1s} \times 100 = 50$$

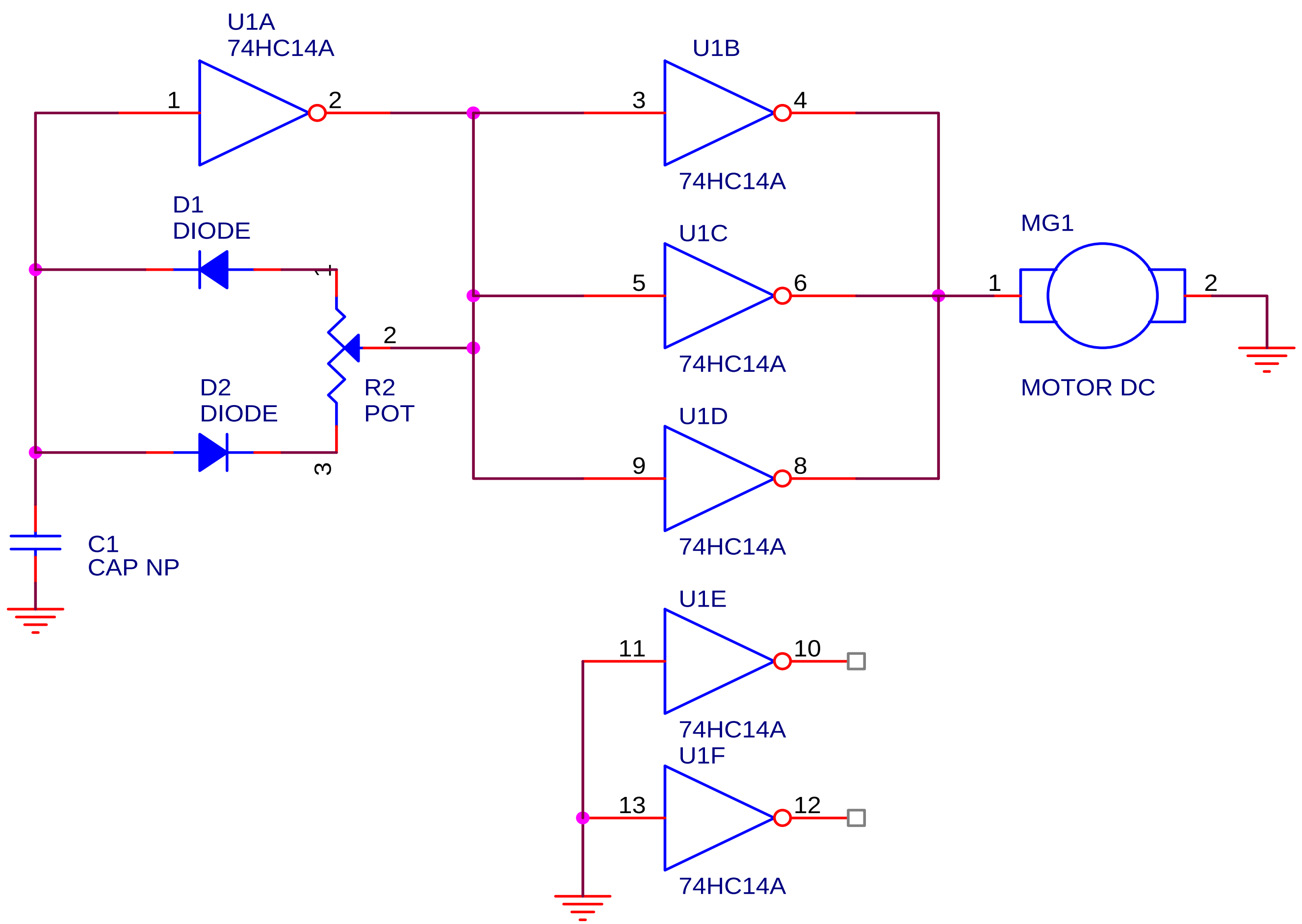
$$\text{Duty Cycle} = 50\%$$



$$\frac{4ms}{5ms} \times 100 = 80$$

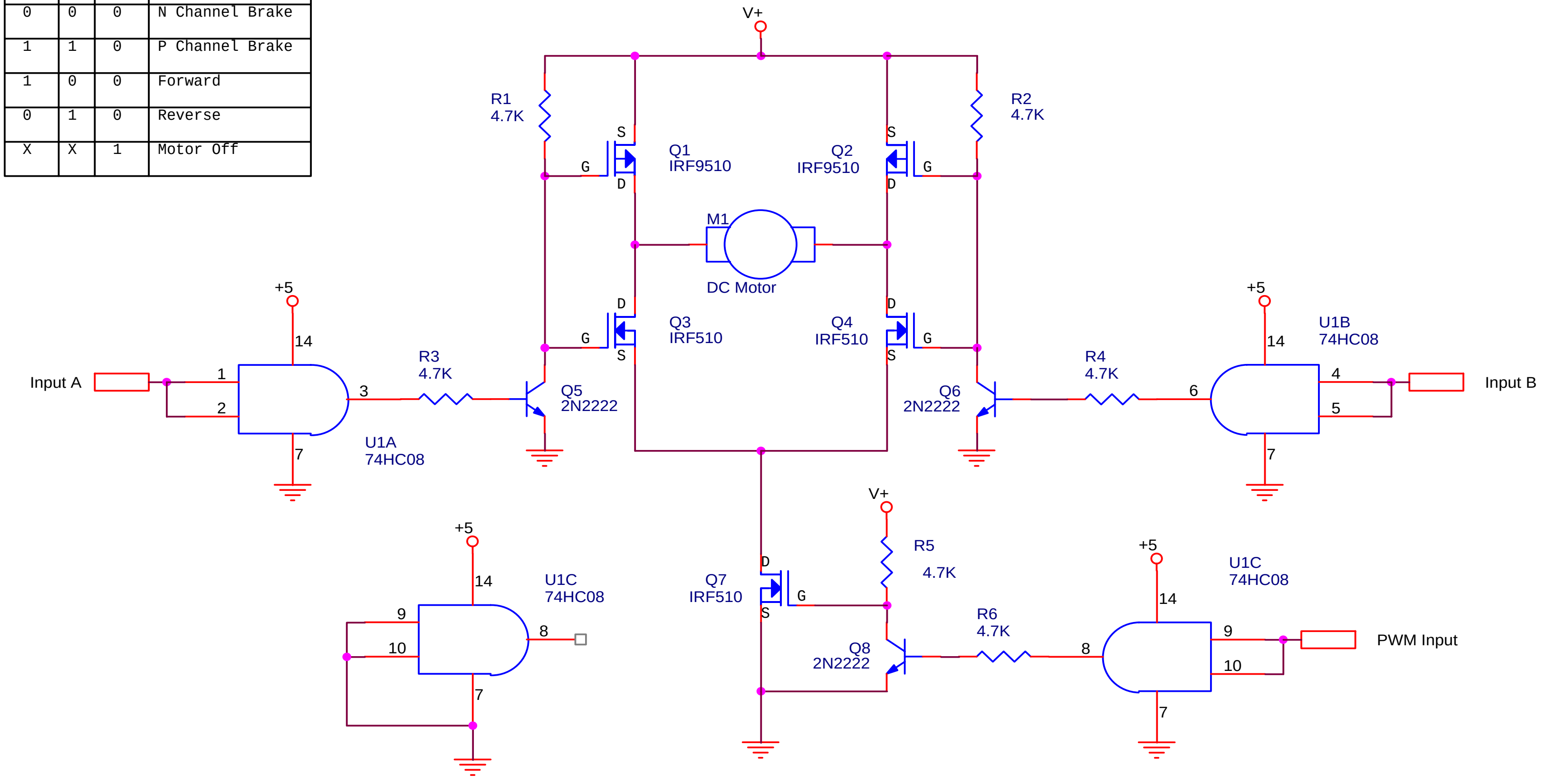
$$\text{Duty Cycle} = 80\%$$

Simple PWM circuit

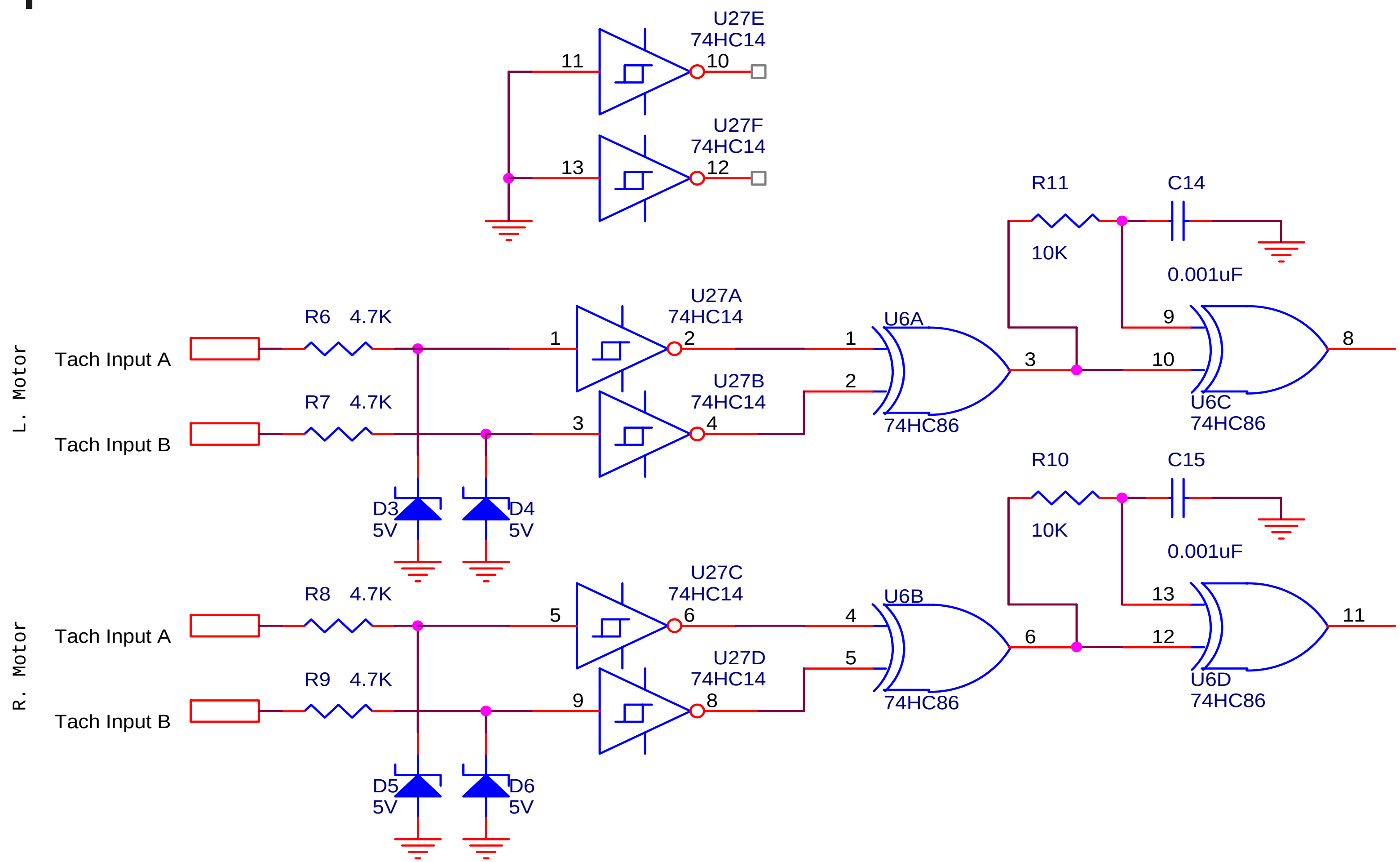


H-Bridge motor driver circuit

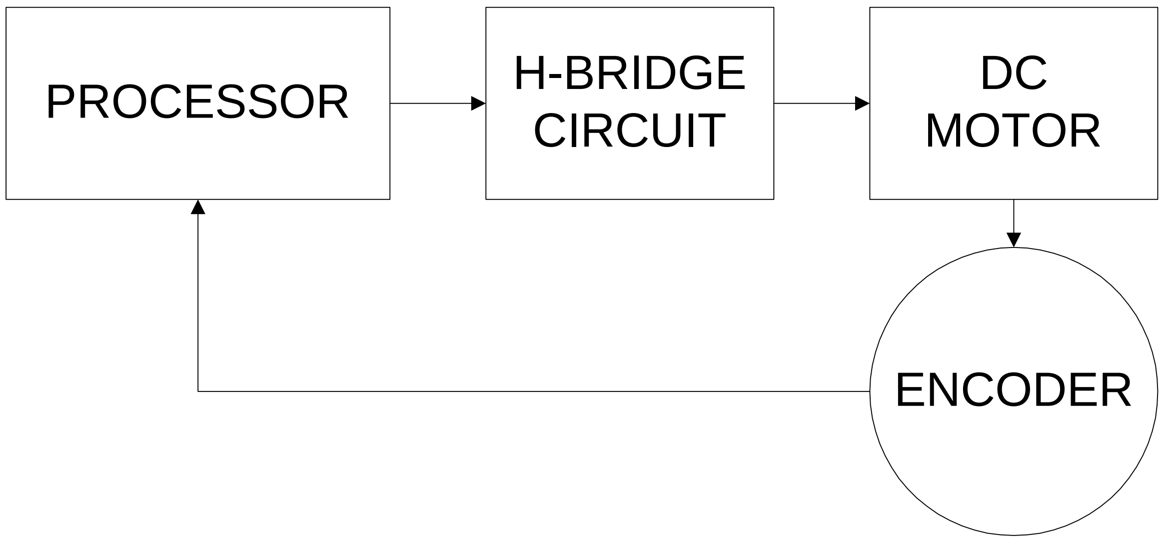
CIRCUIT INPUTS			
A	B	C	Output
0	0	0	N Channel Brake
1	1	0	P Channel Brake
1	0	0	Forward
0	1	0	Reverse
X	X	1	Motor Off



Optical encoder circuit



Motor control diagram



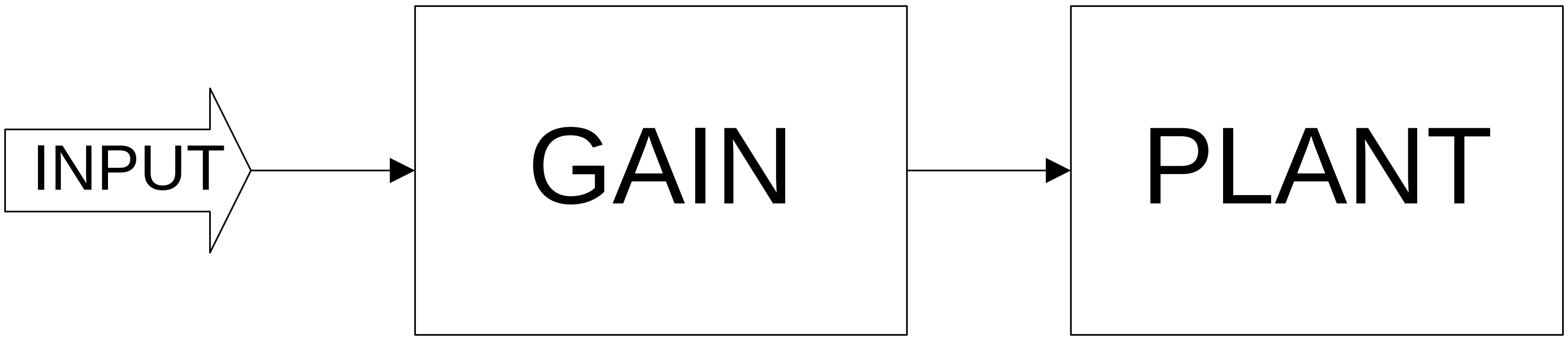
Control systems

- What is a control system?
- What are some examples?
- What are the types of control systems?
- How are control systems represented?

Open loop control systems

- The output of the plant does not affect the input. (No feedback)
- Less common today than closed loop control systems.
- Examples include:
  - Stereo volume control
  - Electric drill speed control

Open loop control system

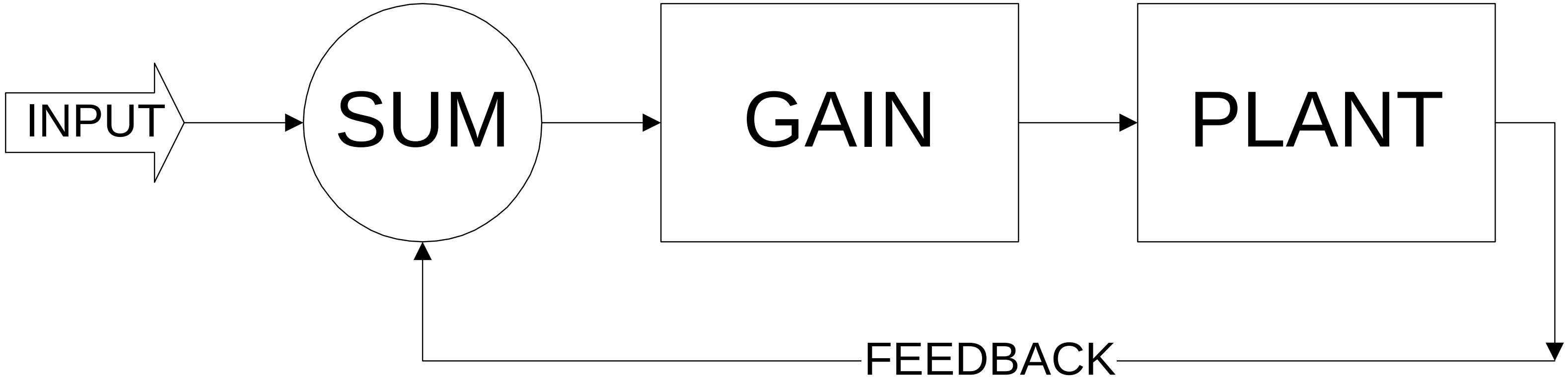


OUTPUT = INPUT X GAIN

Closed loop control systems

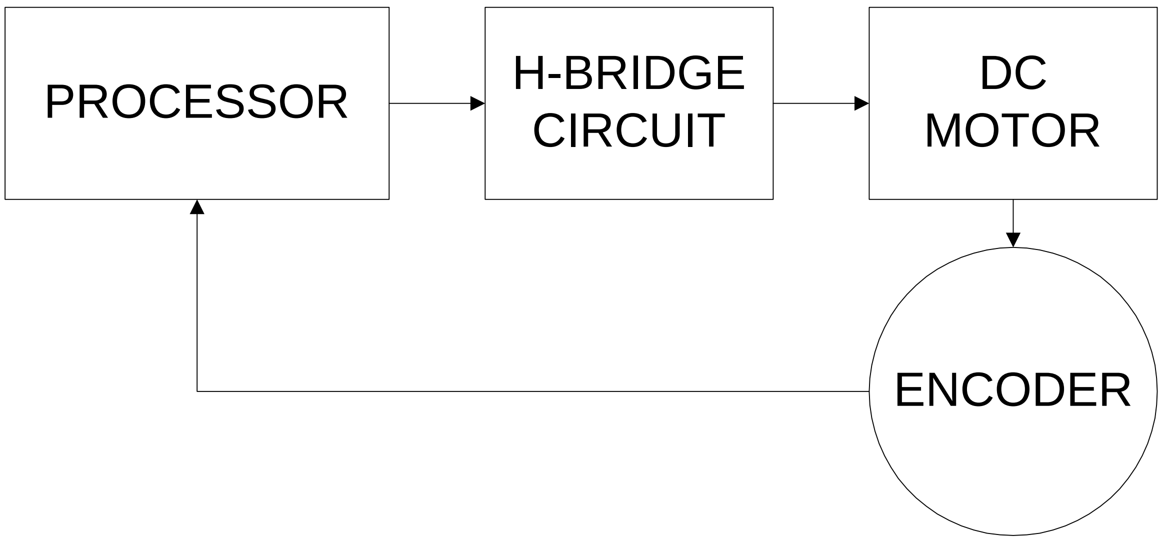
- Use a measurement of output to control the input (Feedback)
- Examples include:
  - Air conditioning thermostat
  - Automobile cruise control

Closed loop control system



OUTPUT = (INPUT – OUTPUT) X GAIN

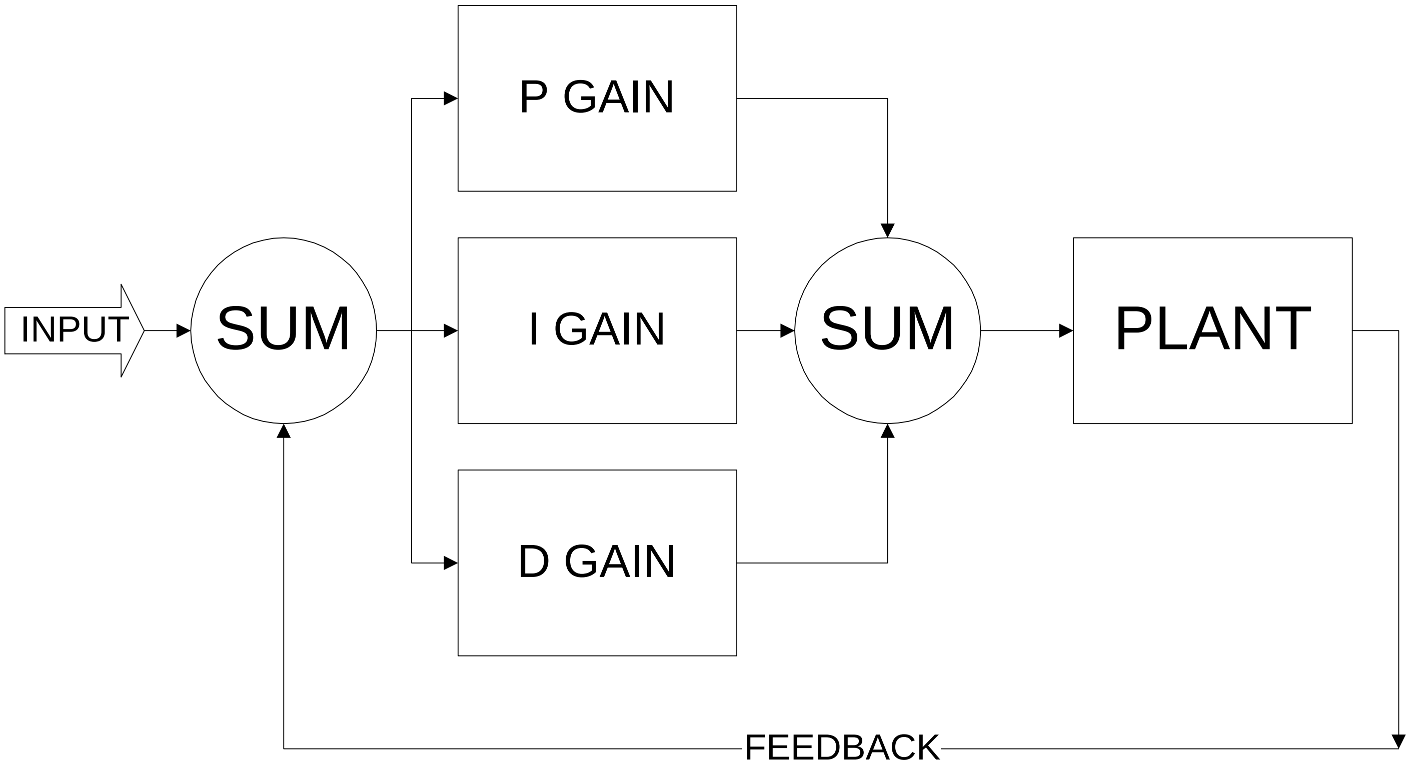
Motor control diagram



PID Closed loop control system

- PID controls the gain portion of the closed loop control system.
- PID algorithms adjust the gain to the plant based on several characteristics of the feedback, not just the current value.

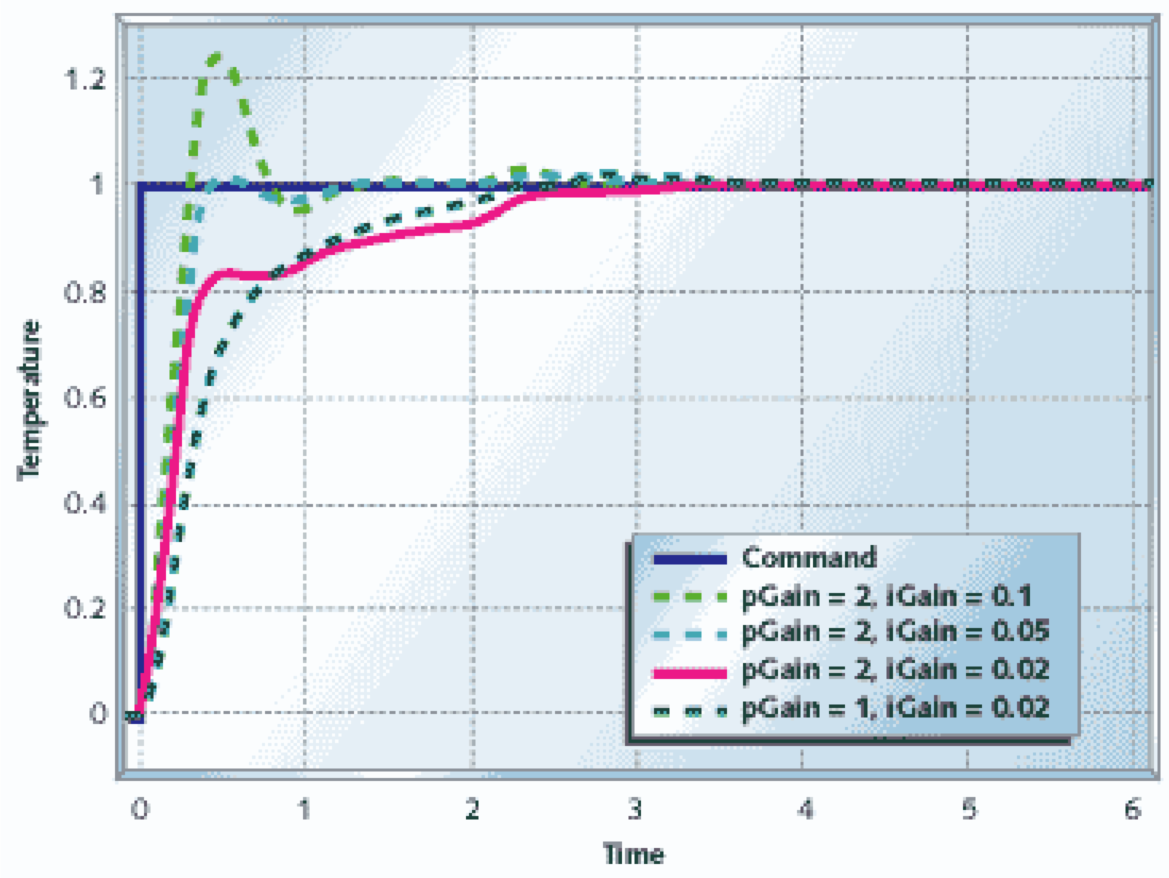
PID control system diagram



OUTPUT = (INPUT – OUTPUT) X (P GAIN + I GAIN + D GAIN)

Sample PID output chart

- Set point
- Rise time
- Overshoot
- Settling time
- Peak time
- Overdamped
- Underdamped



PID implementation

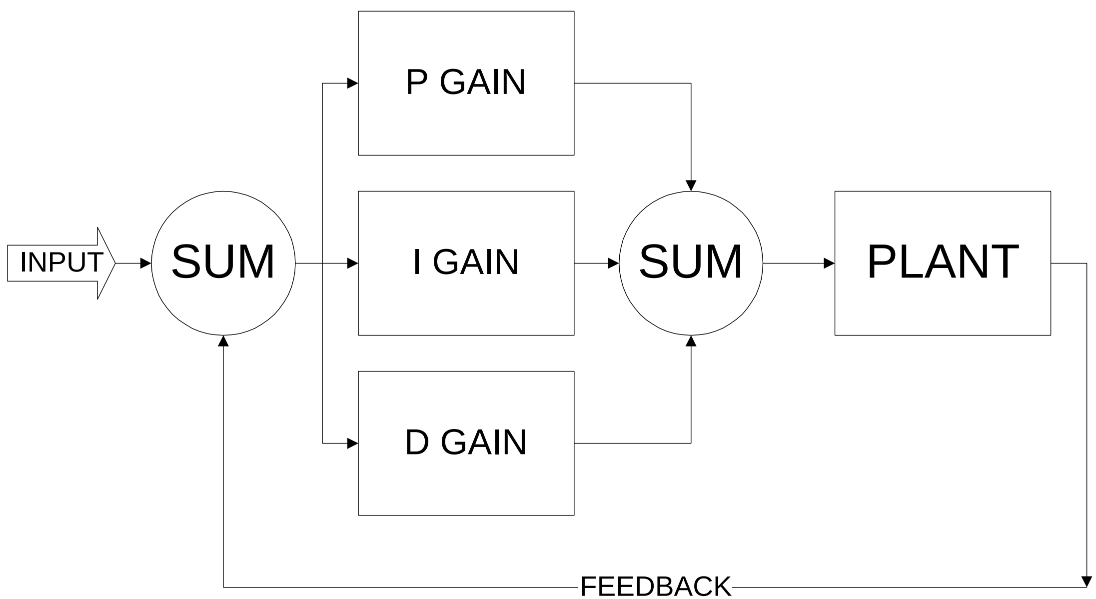
- What is the mathematics of PID?
- How is it programmed?
- What are some common problems?
- How is the PID behavior optimized?

PID variables

- Error term
- P – Proportional gain
- I – Integral gain
- D – Derivative gain

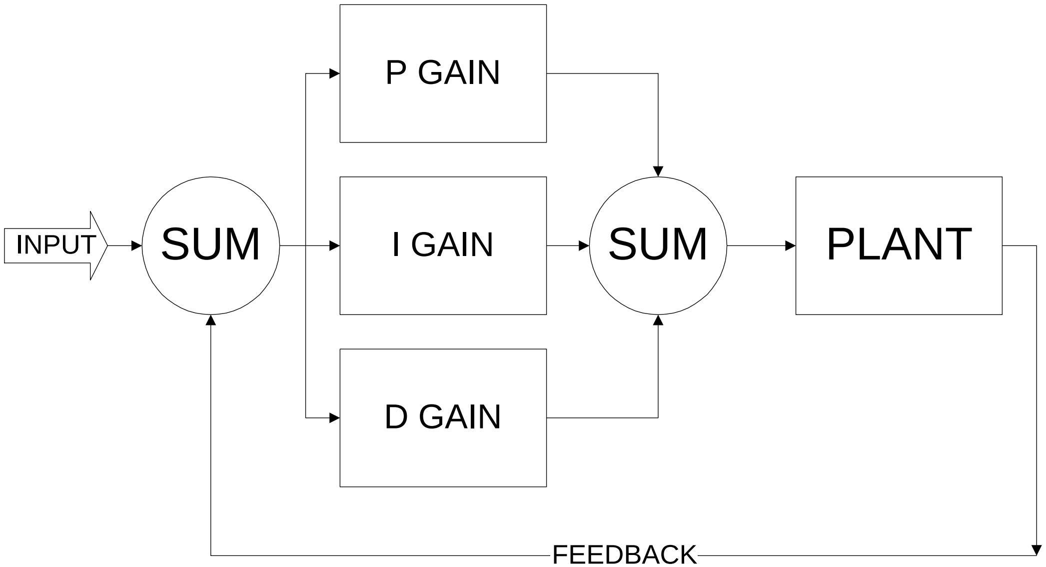
Error term

- The error term is derived by subtracting the feedback (motor speed) from the set point (set speed).
- This is the error in terms of a number of encoder counts per unit time.



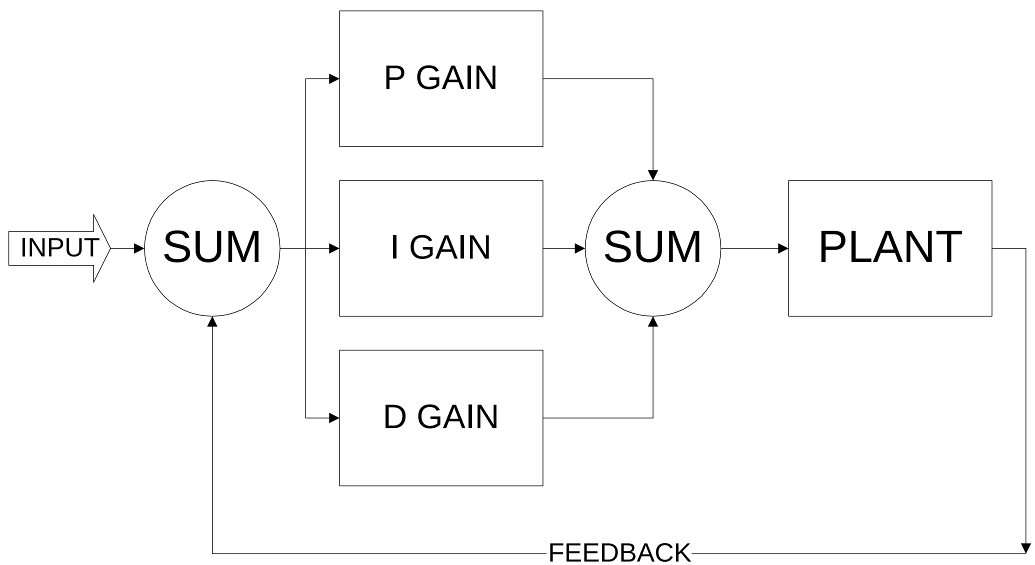
Proportional term

- Simple proportional coefficient Kp is multiplied by the error term.
- Provides linear response to the error term.



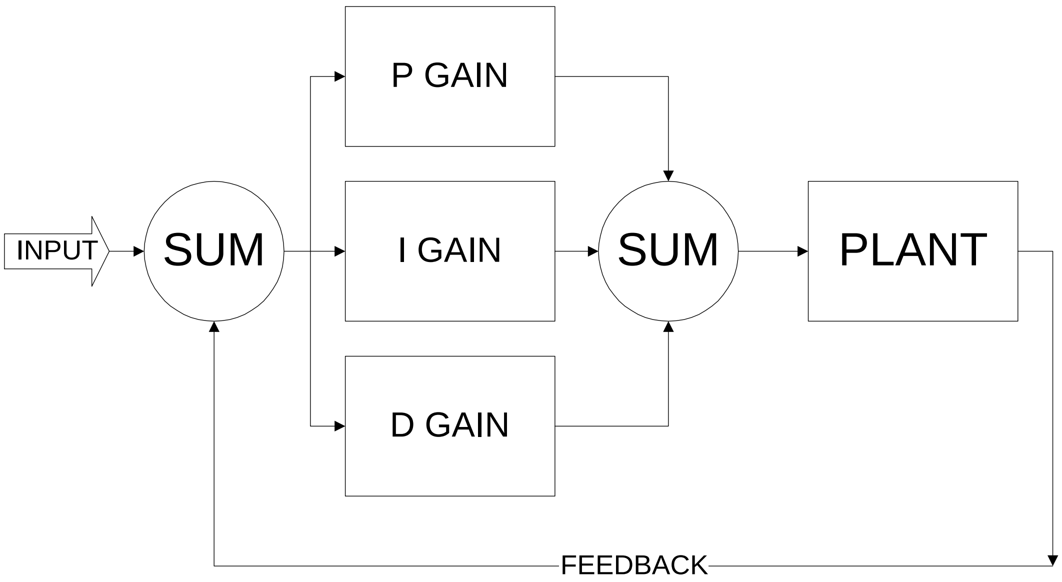
Integral term

- Integral coefficient Ki is multiplied by the error term and added to the sum of all previous integral terms.
- Provides response to accumulated error.



Derivative term

- Derivative coefficient Kd is multiplied by the difference between the previous error and the current error.
- Responds to change in error from one PID cycle to the next.



PID calculation example

- Error_term = Set_Speed – Encoder_Count;
- P_Term = P_Gain * Error_Term;
- D_Term = D_Gain * (Error_Term – D_State);
- D_State = Error_Term;
- I_State = I_State + Error_Term;
- I_Term = I_Gain * I_State;
- PWM_Set = PWM_Set + P_Term + I_Term + D_Term;

Factors to consider

- PID cycle time (0.1 sec)
  - Motor speed (30 rpm)
  - Encoder resolution (500 counts/rev)
  - PWM frequency (1kHz)
- Interrupt driven PID trigger
  - Eliminates code tuning
  - Maintains accurate PID timing

Pitfalls of PID

- Integral windup
- PWM term overflow
- PID variable overflow

Integral windup prevention

```
I_State_L += error_L;             // Accumulate error in I_State
if (I_Term_L > I_Max)             // Check for integral windup
        {
        I_Term_L = I_Max;
        }
else if (I_Term_L < I_Min)
        {
        I_Term_L = I_Min;
        }
```

PWM overflow prevention

```
// *** Set Left PWM ***
fL += P_Term_L + I_Term_L + D_Term_L;        // Set PWM Output
if (fL > 0xFF)                               // Check for PWM Overflow
        {
        fL = 0xFF;                           // Limit fL to prevent windup
        CCAP1H = 0xFF;                       // Set upper limit for PWM Byte
        }
else if (fL < 0x00)
        {
        fL = 0x00;                           // Limit fL to prevent windup
        CCAP1H = 0x00;                       // Set lower limit for PWM byte
        }
else
        {
        CCAP1H = (unsigned char)(fL);
        }
```

PID Tuning

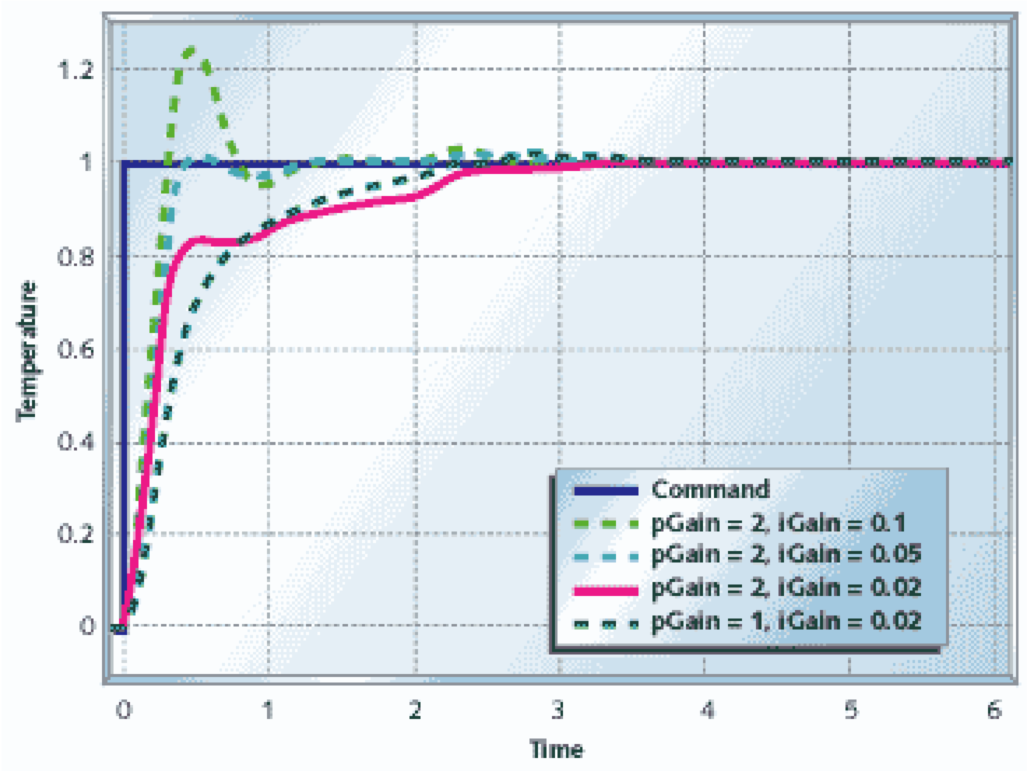
- How is the response of the PID system tested and measured?
- How is the response of the PID system optimized?
- How are the coefficients for P, I, and D determined?

PID tuning (BLACK MAGIC METHODS)

- Mathematical methods
  - Mathematical representation of the plant
  - Root locus methods
  - State space equations
  - Laplace transforms
  - S – domain calculations
  - Is there a simpler way?

PID system measurement

- The behavior of most systems is measured by the system's "Step response"
- How can we measure a step response for our PID controller?



PID Tuning (Brute force approach)

- Add code to monitor the output of the PID algorithm (i.e. encoder speed feedback, counts per PID)
- Store the feedback speed value into an array element for the first 20 PID executions. (2 seconds)
- Change the set speed from 0 to 60% of the motor's maximum speed. (30 counts per PID) This is equivalent to a step function.
- After 2 seconds, stop the motor and print the array data to the serial port.
- This allows the response of the platform to be determined numerically.

PID Brute Force Tuning code

- The tuning algorithm loops through a range of values for each coefficient P, I, and D. For example:

```
for (P=P_start; P<P_end; ++P)
    {
    For (I=I_start; I<I_end; ++I)
        {
        For (D=D_start; D<D_end; ++D)
            {
            Set motor speed to 30 counts/PID
            Wait for the motor to go 1000 counts
            Set motor speed to 0
            Print the P, I, and D values and the 20 array elements
            }
        }
    }
```

PID Tuning (Brute force approach)

Sample of PID tuning data

0,0.001, 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0

0,0.001, 1,8,14,11,7,4,4,1,1,0,0,0,0,0,0,0,0,0,0,0,0

0,0.001, 2,6,15,12,10,9,8,6,7,6,7,6,8,7,7,6,7,7,8,7,8

0,0.001, 3,5,12,12,12,11,11,11,10,10,11,10,11,11,12,11,12,12,11,12,12

0,0.001, 4,6,15,15,14,15,13,13,13,13,14,14,14,14,15,15,14,14,14,14,15

0,0.001, 5,8,16,17,17,16,14,14,14,16,16,16,16,16,18,17,16,15,17,16,15

1,0.001, 0,8,15,12,11,11,14,16,19,27,28,31,32,32,33,33,15,33,33,33,33

1,0.001, 1,5,12,10,11,14,17,21,24,25,27,28,31,31,32,32,32,32,33,32,32

1,0.001, 2,6,13,13,15,15,18,23,24,25,26,28,29,30,31,30,30,31,31,31,31

1,0.001, 3,7,14,16,17,19,20,23,23,25,25,28,29,29,29,30,30,29,31,30,31

1,0.001, 4,6,16,19,18,20,21,23,24,25,26,27,27,28,28,28,29,29,30,29,30

1,0.001, 5,6,18,22,21,22,22,23,23,25,26,27,27,28,28,28,28,29,29,29,30

2,0.001, 0,6,12,12,16,21,27,30,32,34,35,36,35,35,34,32,32,31,30,30,28

2,0.001, 1,6,13,15,19,23,26,29,31,32,34,33,34,0,32,32,32,31,31,30,30

2,0.001, 2,6,14,18,21,24,26,27,30,31,32,32,32,33,32,32,31,31,31,30,29

2,0.001, 3,5,18,21,23,26,27,28,30,29,29,29,28,28,30,29,30,31,31,31,32

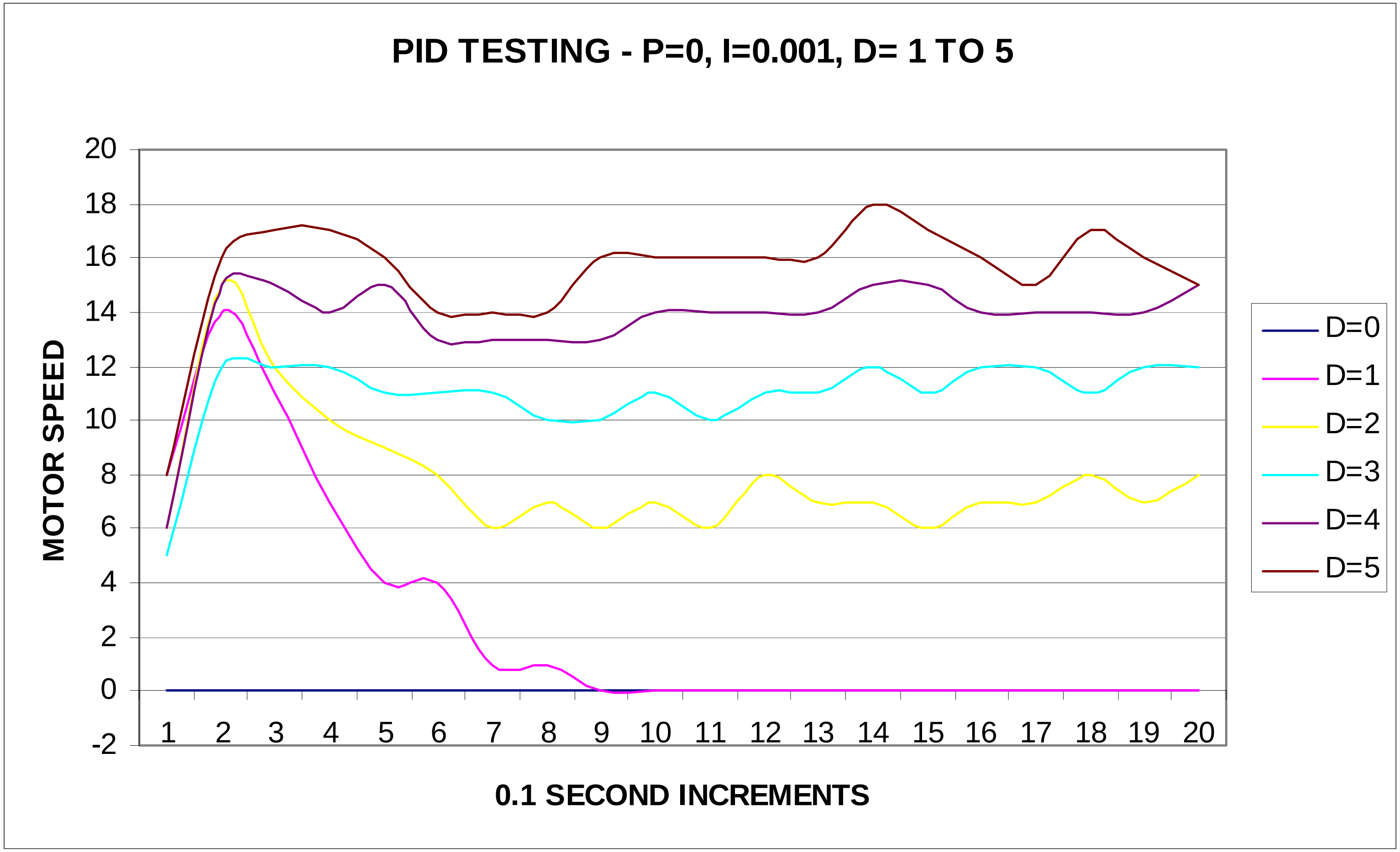
2,0.001, 4,6,18,24,26,25,25,26,27,30,30,30,30,31,31,31,30,30,31,30,30

2,0.001, 5,6,19,27,26,26,26,26,28,29,30,30,30,30,29,31,30,30,30,30,31

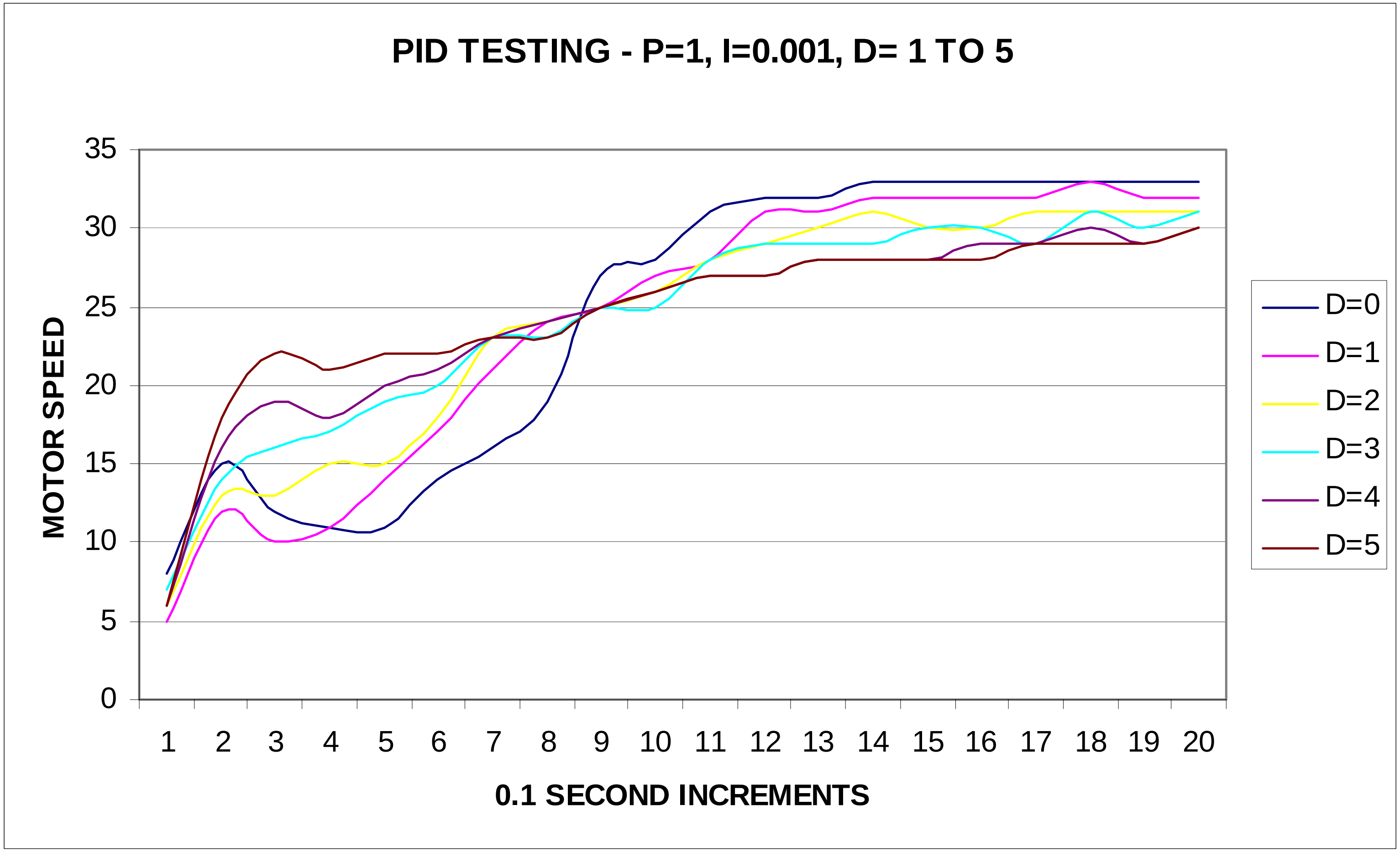
PID Brute Force Tuning results

- Now the results of all PID values within the test range are plotted with respect to time.
- The values which yield the best curve will be used for the PID controller.

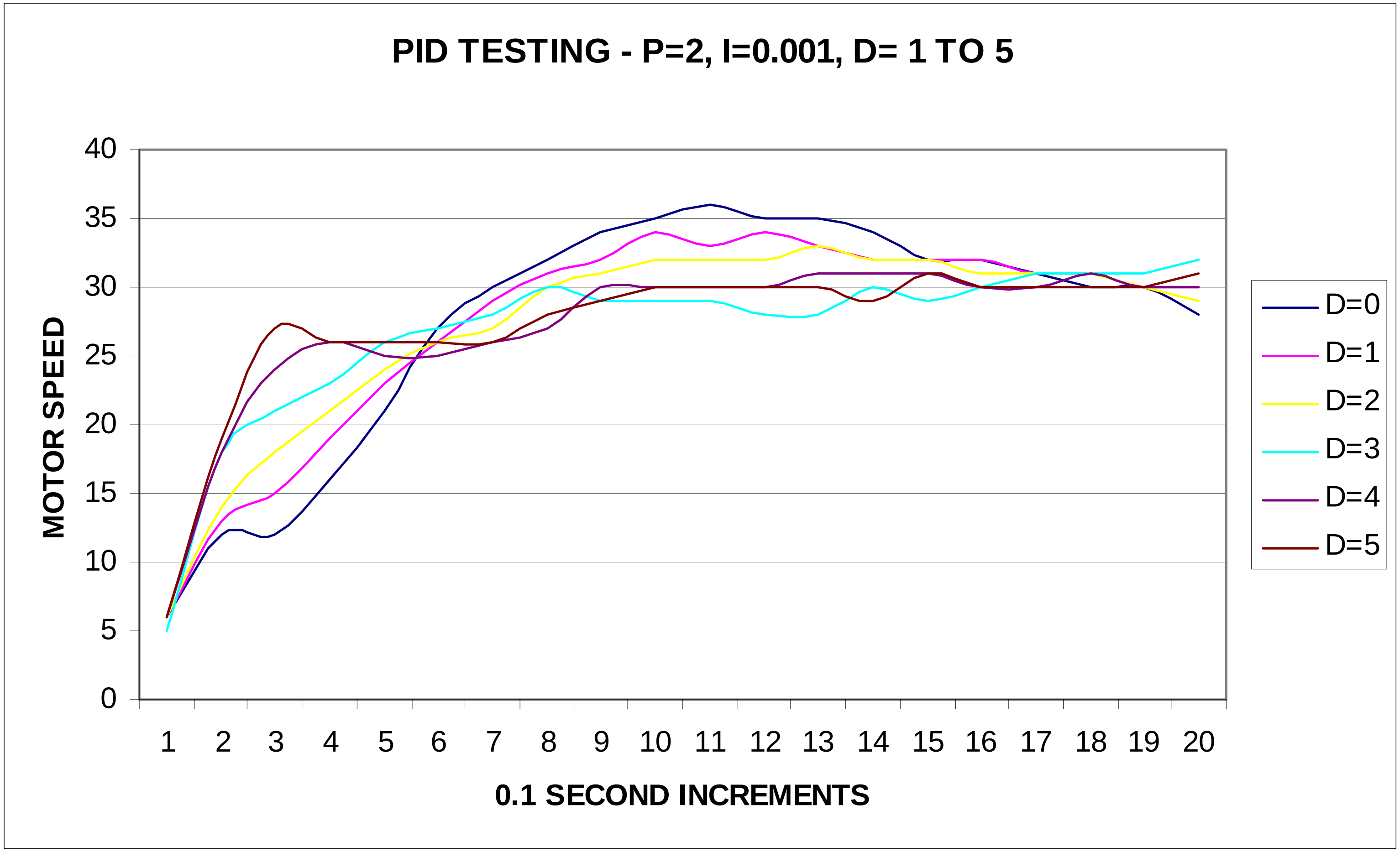
PID Tuning chart 1



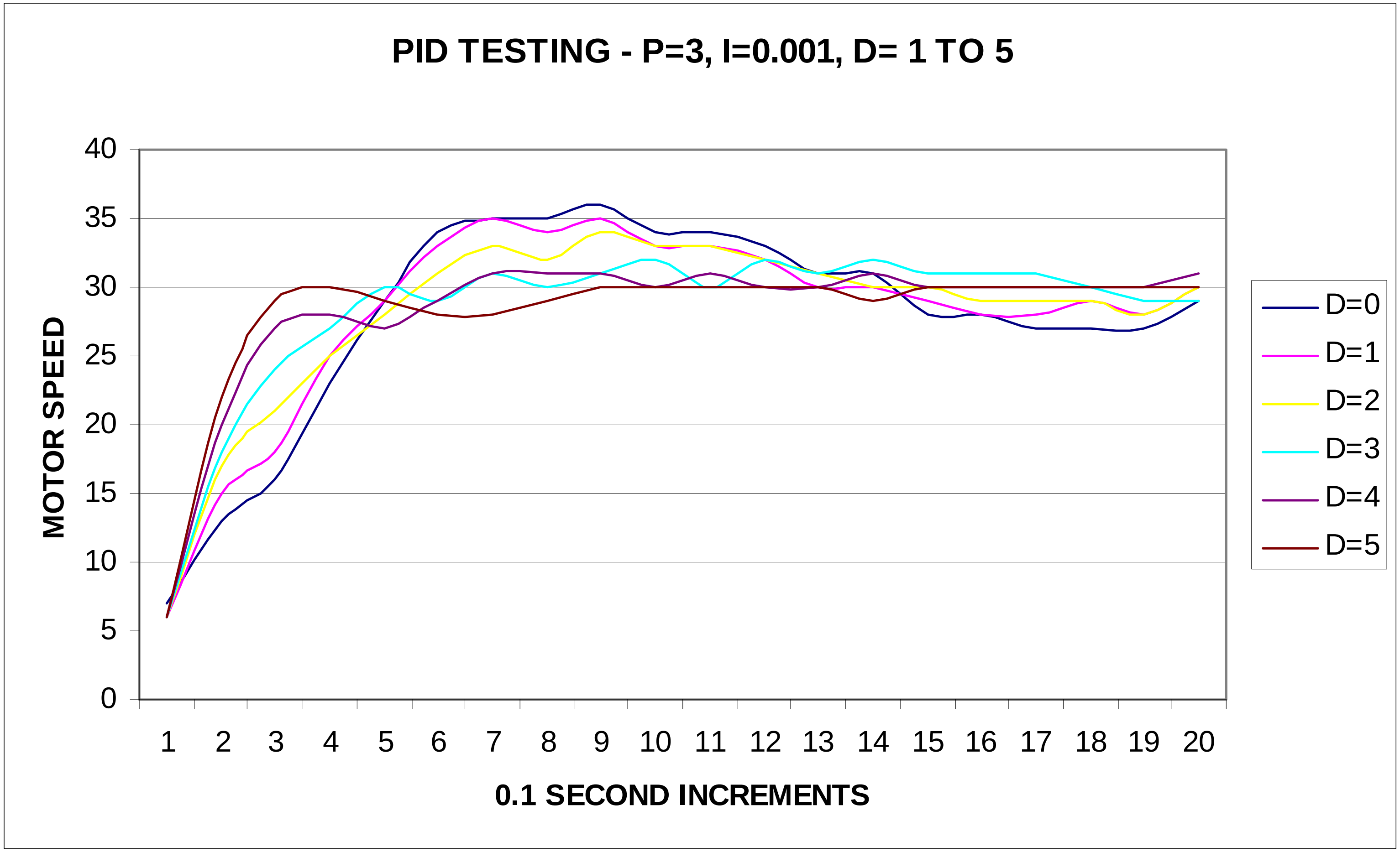
PID Tuning chart 2



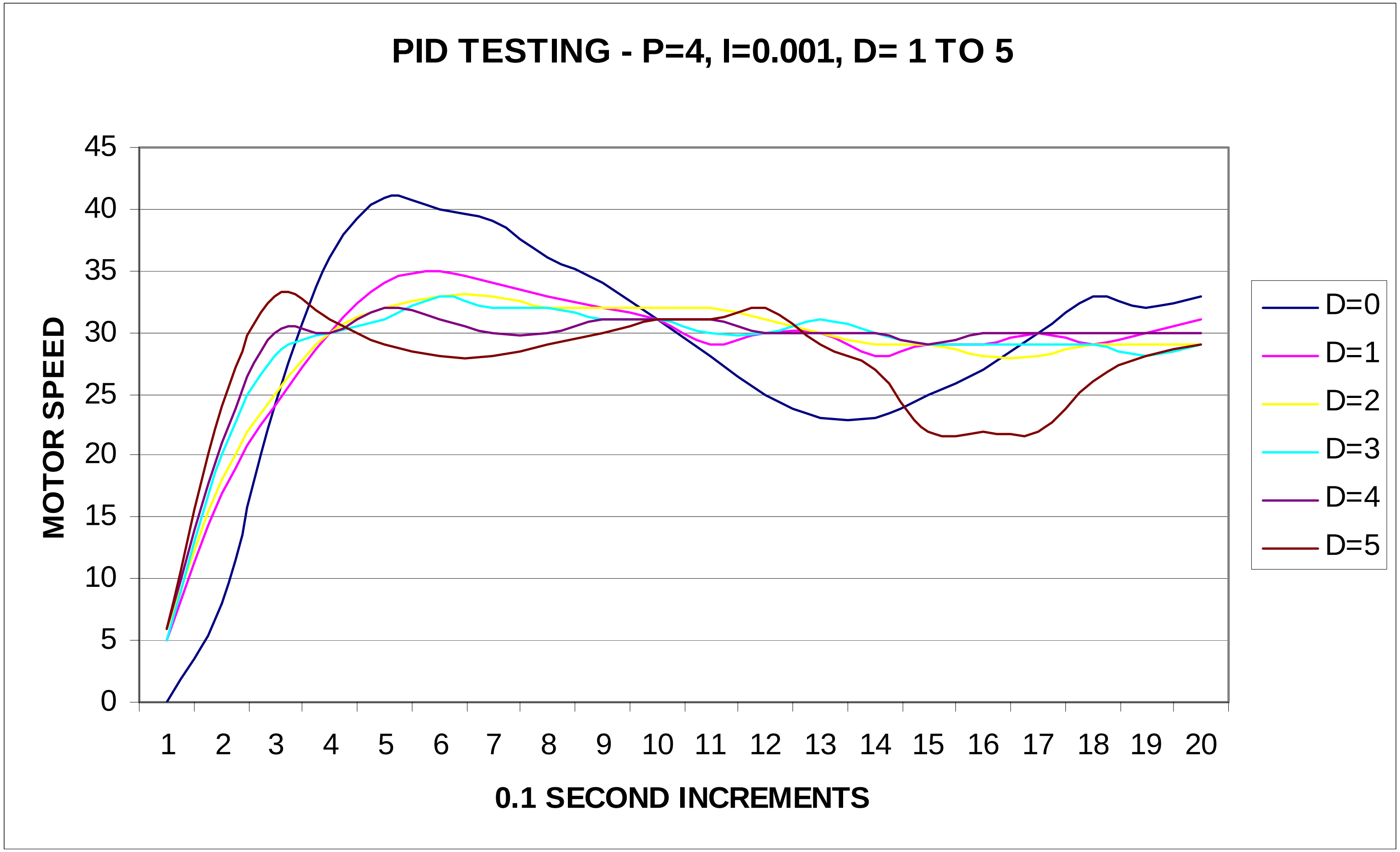
PID Tuning chart 3



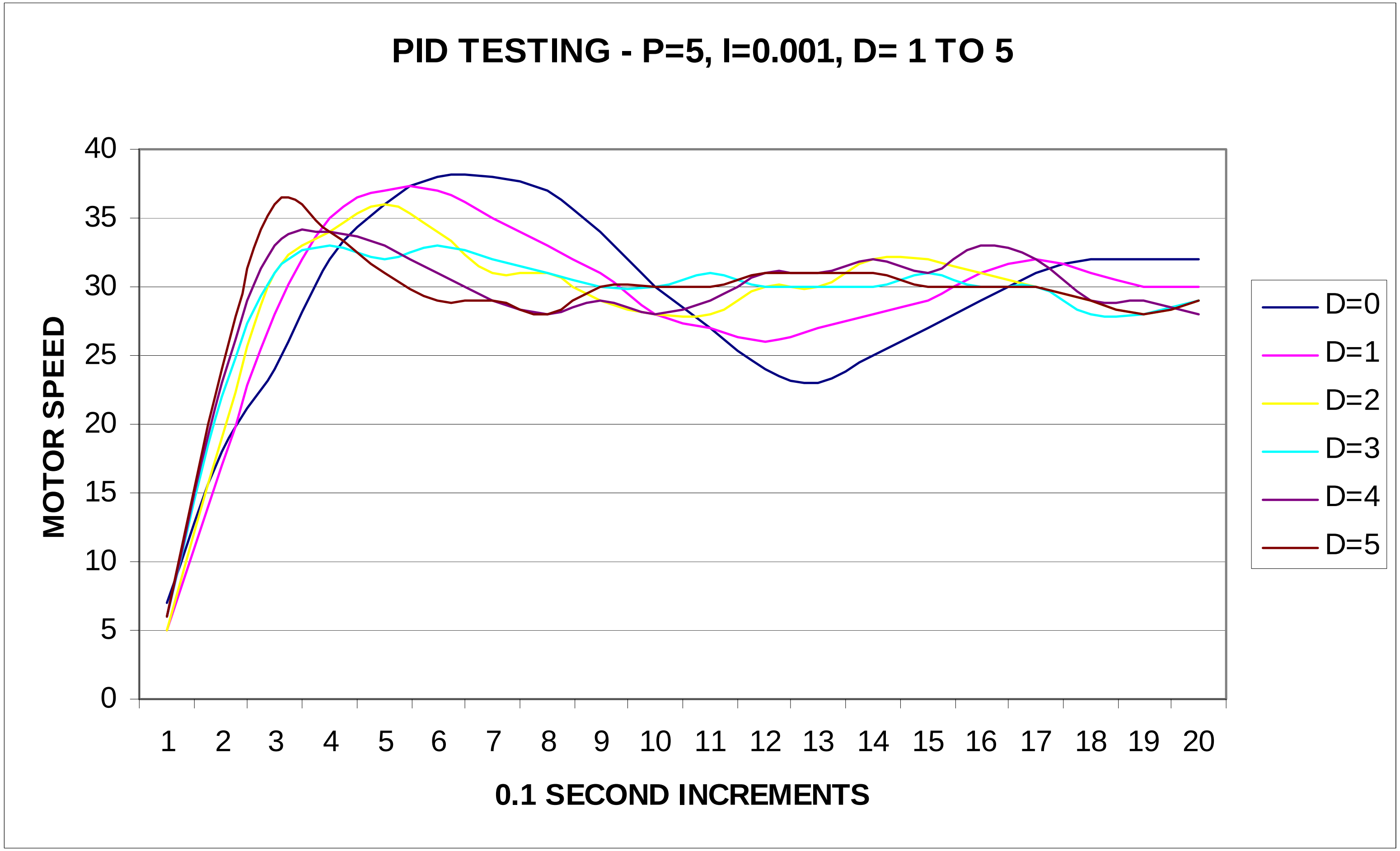
PID Tuning chart 4



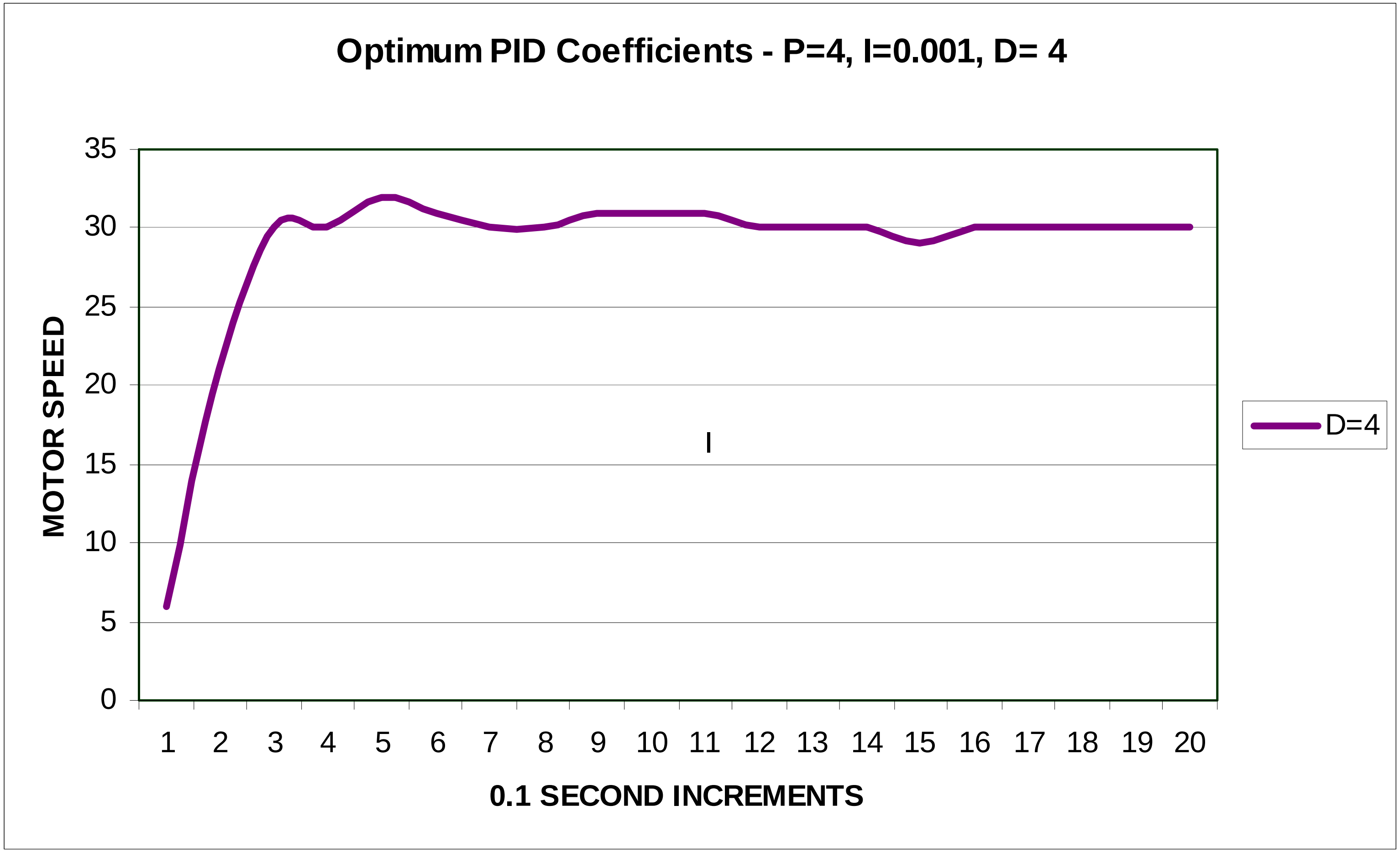
PID Tuning chart 5



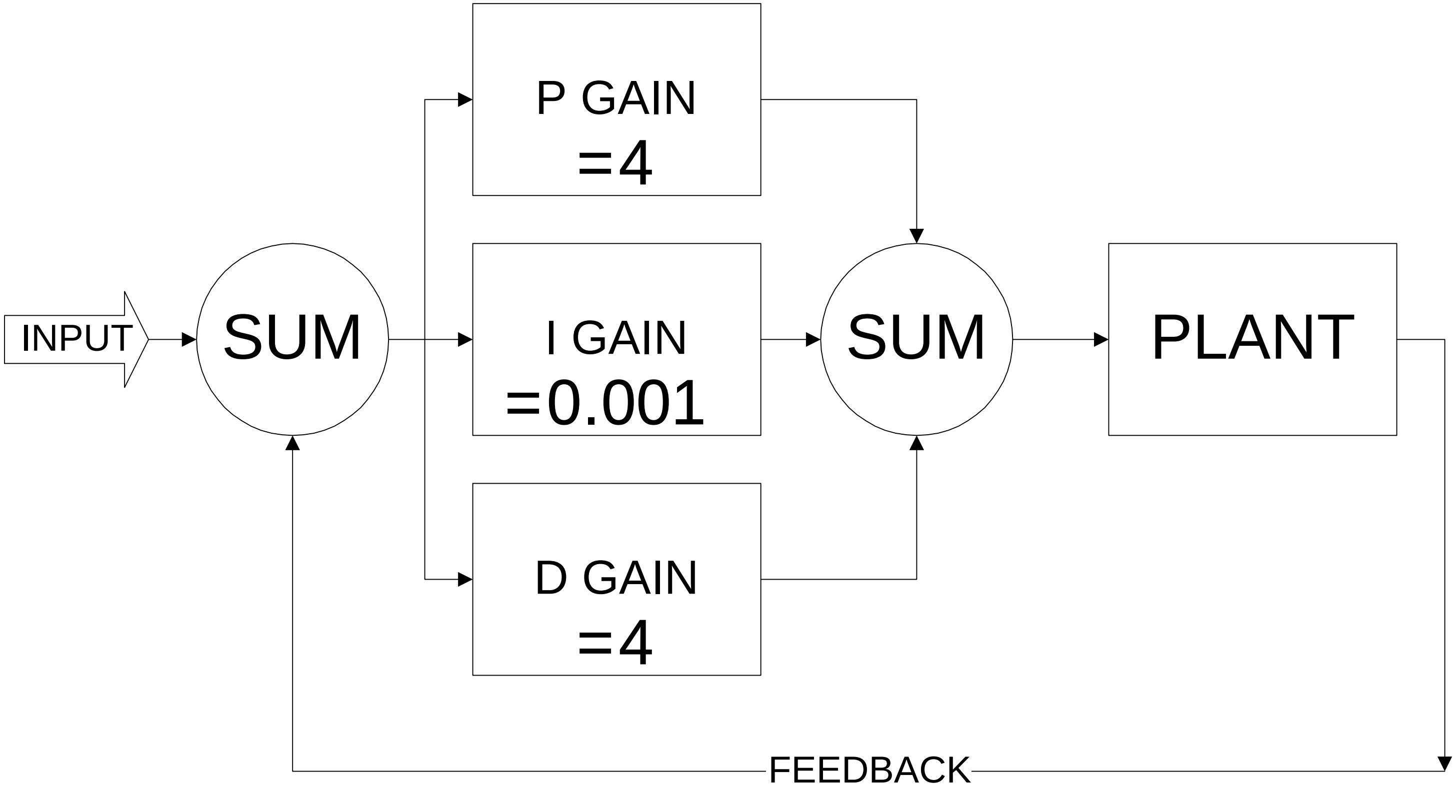
PID Tuning chart 6



The optimum PID coefficients!



The completed control system



PID References

- Carnegie Mellon University
  http://www.engin.umich.edu/group/ctm/PID/PID.html
- “PID Without a PHD” by Tim Wescott
  http://www.embedded.com/2000/0010/0010feat3.htm

This concludes the presentation

Thank you for your attention