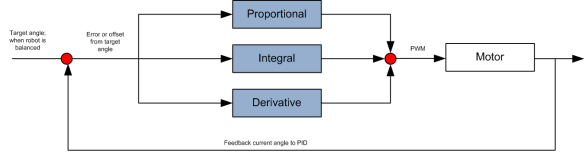
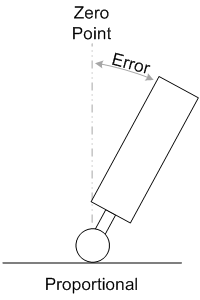
**Proportional Integral Derivative (PID)**

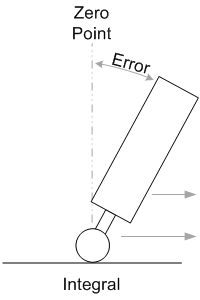
A PID control algorithm is used to balance the robot by driving the motors based on the tilt.

[](http://i2.wp.com/ozzmaker.com/wp-content/uploads/2013/04/pid.png)

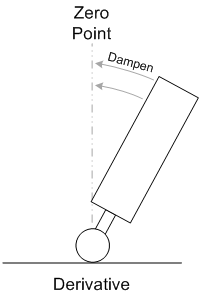
**The proportional(P)** term of the PID is based on the current angle difference or error from Point Zero (where robot balances) multiplied by the P gain. The P gain is a number we need to tweak to get the right amount of proportional control.  
The proportional control will try and correct the balance based on the current error from point zero.



The portion of code for the P term is;  
  
Pterm = KP \* CFangleX;  
  
**The integral (I)** term of the PID is based on the current angle difference or error from Point Zero multiplied by the I gain, which is that accumulated over time. The integral control assist in balancing the robot if it is moving.



The portion of code for the I term is;  
  
iTerm += KI \* CFangleX;  
  
**The derivative (D)** term of the PID is based on the current rate of rotation. It is used to dampen the response as the robot reaches Zero Point. Without the derivative control, the robot will overshoot and than start to oscillate.



The portion of code for the D term is;  
  
dTerm = KD \*  (CFangleX – lastAngle);  
lastAngle = CFangleX;  
  
  
The addition of all the PID values will be used to drive the motors in the right direction and with the right power;  
  
output = Pterm + iTerm + dTerm;  
  
Getting the PID values correct(KP,KI,KD) is the hardest part of building PiBBOT.  
  
The PID values need to be manually tuned;

1. Set KI and KD to zero.
2. Set KP high enougth so that the motors drive the wheels under the robot in the direction it is falling. The robot should overshoot Point Zero a little bit and then start to oscillate.
3. Reduce KP by about 10% to so you get just below the oscillation.
4. Increase KI. This will help the robot reach Point Zero faster and will also start to oscillate. Try and get a value so that the robot just oscillates
5. Increase KD to dampen the oscillation and until the robot balances.

It is best to change these values while trying to balance to see the results in real time, a potentiometer could be used for this. I have used the RF module to increase or decrease the values.