

Project Blog

Project updates and... um...

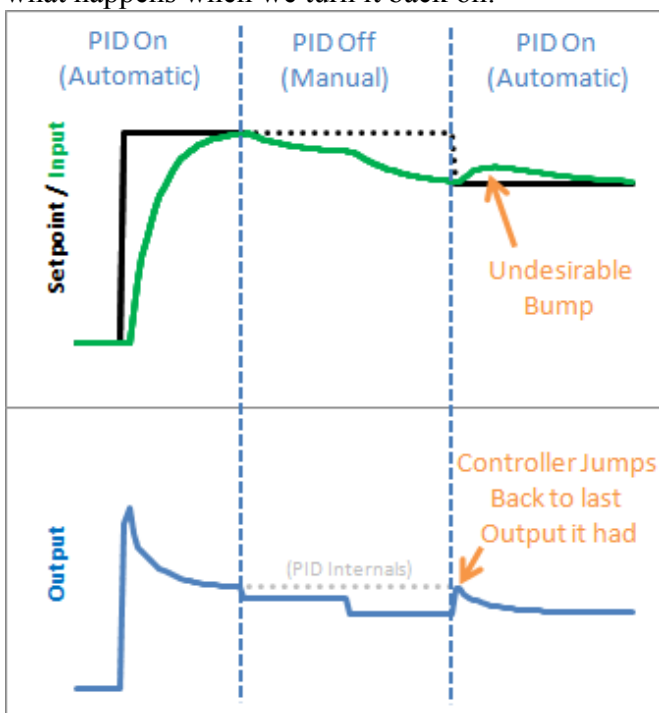
« [Improving the Beginner's PID: On/Off](#)
[Improving the Beginner's PID: Direction](#) »

Improving the Beginner's PID: Initialization

(This is Modification #6 in a [larger series](#) on writing a solid PID algorithm)

The Problem

In the last section we implemented the ability to turn the PID off and on. We turned it off, but now let's look at what happens when we turn it back on:



Yikes! The PID jumps back to the last Output value it sent, then starts adjusting from there. This results in an Input bump that we'd rather not have.

The Solution

This one is pretty easy to fix. Since we now know when we're turning on (going from Manual to Automatic,) we just have to initialize things for a smooth transition. That means massaging the 2 stored working variables (ITerm & lastInput) to keep the output from jumping.

The Code

```
1  /*working variables*/
2  unsigned long lastTime;
3  double Input, Output, Setpoint;
4  double ITerm, lastInput;
5  double kp, ki, kd;
6  int SampleTime = 1000; //1 sec
7  double outMin, outMax;
8  bool inAuto = false;
```

```
9
10 #define MANUAL 0
11 #define AUTOMATIC 1
12
13 void Compute()
14 {
15     if(!inAuto) return;
16     unsigned long now = millis();
17     int timeChange = (now - lastTime);
18     if(timeChange>=SampleTime)
19     {
20         /*Compute all the working error variables*/
21         double error = Setpoint - Input;
22         ITerm+= (ki * error);
23         if(ITerm> outMax) ITerm= outMax;
24         else if(ITerm< outMin) ITerm= outMin;
25         double dInput = (Input - lastInput);
26
27         /*Compute PID Output*/
28         Output = kp * error + ITerm- kd * dInput;
29         if(Output> outMax) Output = outMax;
30         else if(Output < outMin) Output = outMin;
31
32         /*Remember some variables for next time*/
33         lastInput = Input;
34         lastTime = now;
35     }
36 }
37
38 void SetTunings(double Kp, double Ki, double Kd)
39 {
40     double SampleTimeInSec = ((double)SampleTime)/1000;
41     kp = Kp;
42     ki = Ki * SampleTimeInSec;
43     kd = Kd / SampleTimeInSec;
44 }
45
46 void SetSampleTime(int NewSampleTime)
47 {
48     if (NewSampleTime > 0)
49     {
50         double ratio = (double)NewSampleTime
51                        / (double)SampleTime;
52         ki *= ratio;
53         kd /= ratio;
54         SampleTime = (unsigned long)NewSampleTime;
55     }
56 }
57
58 void SetOutputLimits(double Min, double Max)
59 {
60     if(Min > Max) return;
61     outMin = Min;
62     outMax = Max;
63
64     if(Output > outMax) Output = outMax;
65     else if(Output < outMin) Output = outMin;
66
67     if(ITerm> outMax) ITerm= outMax;
68     else if(ITerm< outMin) ITerm= outMin;
69 }
70
```

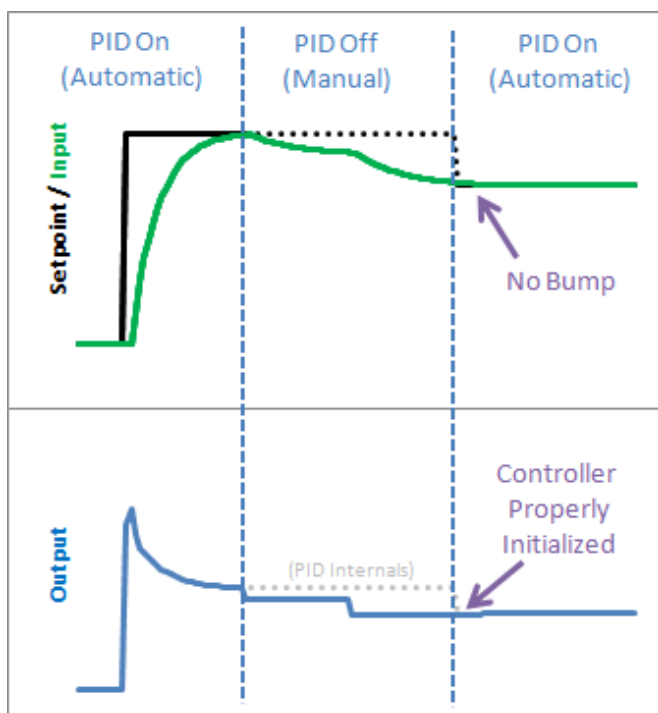
```

71 void SetMode(int Mode)
72 {
73     bool newAuto = (Mode == AUTOMATIC);
74     if(newAuto && !inAuto)
75     { /*we just went from manual to auto*/
76         Initialize();
77     }
78     inAuto = newAuto;
79 }
80
81 void Initialize()
82 {
83     lastInput = Input;
84     ITerm = Output;
85     if(ITerm > outMax) ITerm = outMax;
86     else if(ITerm < outMin) ITerm = outMin;
87 }

```

We modified SetMode(...) to detect the transition from manual to automatic, and we added our initialization function. It sets ITerm=Output to take care of the integral term, and lastInput = Input to keep the derivative from spiking. The proportional term doesn't rely on any information from the past, so it doesn't need any initialization.

The Result



We see from the above graph that proper initialization results in a bumpless transfer from manual to automatic: exactly what we were after.

[Next >>](#)

Update: Why not ITerm=0?

I have been getting a lot of questions recently asking why I don't set ITerm=0 upon initialization. As an answer, I'd ask you to consider the following scenario: The pid is in manual, and the user has set the output to 50. After a time, the process steadies out to an input of 75.2. The user makes the Setpoint 75.2 and turns on the pid. What should happen?

I contend that after switching to automatic the output value should stay at 50. since the P and D terms will be zero, the only way this will happen is if ITerm is initialized to the value of Output.

If you are in a situation where you need the output to initialize to zero, there is no need alter the code above. Just set Output=0 in your calling routine before turning the PID from Manual to Automatic.



Tags: [Arduino](#), [Beginner's PID](#), [PID](#)

This entry was posted on Friday, April 15th, 2011 at 3:06 pm and is filed under [Coding](#), [PID](#). You can follow any responses to this entry through the [RSS 2.0](#) feed. You can [leave a response](#), or [trackback](#) from your own site.

8 Responses to “Improving the Beginner’s PID: Initialization”

1.  *Paul* says:

[March 8, 2012 at 3:21 pm](#)

Why do you set the Iterm=output?

2.  *wezzo* says:

[May 4, 2012 at 9:29 pm](#)

This is a really fantastic blog that gives a really good balance of practical problem->solution stuff and enough background to give it context.

Really very well done!

I’m still geeking out, and my inner nerd has ‘horny rimmed glasses’ over this. I jest ye not!

Hoping to adapt this to Arduino PID temp controller and maybe notso wobbly wobblebot.

3.  *Eugeniu* says:

[June 3, 2012 at 5:20 am](#)

Thanks. This series of posts helped me a lot in developing pid controllers for freescale car competition.

4.  *Juan* says:

[November 7, 2012 at 3:33 am](#)

I have the same question as Paul; why do you set Iterm = Output if Output has not been updated since you were last in manual mode?

Shouldn’t Output = 0 and then start integrating as soon as the next error is calculated on the new output?

Thanks for putting in the effort of explaining PID in such an intuitive way. I have been on control courses which leave you with less understanding!

5.  *Mike* says:


[November 14, 2012 at 3:13 pm](#)

It looks like if SetMode(AUTOMATIC) is repeatedly called the PID will continually initialize. I need to add a variable to detect if the previous Mode was MANUAL and only then initialize.

6.  *Mike* says:


[November 14, 2012 at 3:22 pm](#)

Nevermind, needed to read one line down. I see where you are detecting the transition.

7.  *mecharobo* says:
[September 7, 2013 at 12:39 pm](#)

he used $item = output$, because the integration part is the most affecting part and others affect much much less.

Note: the output IS the last manual desired setpoint, so the last output is chosen not from the pid equation but from the user.

8.  *Raul* says:
[April 19, 2016 at 1:44 pm](#)

I just ported your code to Javascript and can be used in this online/interactive demo

<http://codinglab.blogspot.com/2016/04/online-pdi-trainer.html>

Leave a Reply

<input type="text"/>	Name (required)
<input type="text"/>	Mail (will not be published) (required)
<input type="text"/>	Website

Submit Comment

- Search for:

• Links





• This Site

- [About](#)
- [Project Index](#)

• Categories

- [PID](#) (23)
 - [Coding](#) (11)
 - [Front End](#) (2)
 - [Showcase](#) (4)
- [Projects](#) (40)
 - [Craft](#) (6)
 - [Electronic](#) (12)
 - [Mechanical](#) (26)
- [Uncategorized](#) (5)

• Archives

- [November 2012](#)
- [September 2012](#)
- [July 2012](#)
- [June 2012](#)
- [April 2012](#)
- [March 2012](#)
- [January 2012](#)
- [December 2011](#)
- [October 2011](#)
- [September 2011](#)
- [August 2011](#)
- [July 2011](#)
- [June 2011](#)
- [May 2011](#)
- [April 2011](#)
- [September 2010](#)
- [August 2010](#)
- [July 2010](#)
- [March 2010](#)
- [November 2009](#)
- [October 2009](#)
- [September 2009](#)
- [August 2009](#)
- [July 2009](#)
- [June 2009](#)
- [May 2009](#)

Project Blog is proudly powered by [WordPress](#)
[Entries \(RSS\)](#) and [Comments \(RSS\)](#).