If this is your first visit, be sure to check out the **FAQ** by clicking the link above. You may have to **register** before you can post: click the register link above to proceed. To start viewing messages, select the forum that you want to visit from the selection below.

If you'd like to receive bonuses in the Official Wiremod Gmod Server, Link your Steam account to you forum account **here!**

Thread: PID controller - what is it and how to?

12-13-2009 #1

Fizyk

Wire Sofaking

WIREMOD MEMBER

Join Date: Jun 2008
Location: Łomianki, Poland
Posts: 744
Blog Entries: 1

PID controller - what is it and how to?

I see many people still don't know this excellent and pretty simple method of stabilizing things, so I decided to write a tutorial about it.

What is it?

PID controller is a thing that lets you stabilize many different things. The most basic example is a prop hovering in place - you take its position, feed it to a PID controller, and the controller outputs a force that can be used to hold the prop in place.

How does it work?

A PID controller is used to make a quantity (like position) reach a target value (a target position). The first thing a PID controller does is to calculate the **error**. The error is just the difference between the target value and the current value, for example:

Code:

```
Error = TargetPos - E:pos()
```

Having the error, PID generates three values based on it, that the letters "P", "I" and "D" stand for:

- **P** - **proportional** term - it's just Number*Error, which means the greater the error, the greater the P term.
- **I** - **integral** term - this is more complicated. It's a value that comes from summing errors over time. Basically,

the greater the error, the faster this value grows. For those of you familiar with calculus, it's just Error integrated over time.
- **D** - **derivative** term - value that measures the change of Error over time. In GMod it will usually be just delta of Error, or $Error in E2 - that is, if at one moment the error is Error1, and a while later it's Error2, then the D term will equal Error2-Error1. Sometimes it's good to divide it by change (or delta) of time. For those familiar with calculus, it's just derivative of Error over time.

P, I and D terms can be generated in E2 like that:

Code:

```
@inputs TargetPos:vector
@persist T [ITerm Error]:vector

runOnTick(1)

T = curtime()
Error = TargetPos - entity():pos() #calculate error

#initialize gains
GainP = SomeNumber
GainI = SomeNumber
GainD = SomeNumber

PTerm = GainP * Error
ITerm += GainI * Error * $T
DTerm = GainD * $Error / $T
```

Gains describe influence of the terms on the output value. The greater the gain, the more influence on output will a term have. Gain equal to 0 makes a term equal to 0 too, so it will have no influence on the output at all then.

Now, when we have P, I and D terms, we can generate output - it's just sum of P, I and D terms:

Code:

```
Out = PTerm + ITerm + DTerm
```

More detailed description of the P, I, D terms

Ok, we know what are the P, I, D terms and how to calculate them, but... what do they do?

P term can be thought of like that: when you use the PID controller to hold a prop in position, P term makes the force grow with distance from target. The further away from target the prop is, the stronger it is pushed towards the target. When it is exactly in place, the P term is 0, so it isn't pushed at all. Great!

But... When the prop was far away, it was pushed towards the target all the time along its way to the target. So, when it reaches the desired position, the force is 0, but it has a nonzero velocity... Simply put, it is still flying quite fast, so it will miss the target, turn around at some place and head to the target again... and again miss it. Here the D term comes to help!

The D term causes the prop to brake harder when it goes faster. This way, when the prop is going towards the target, the P term is pushing it to go faster, and the D term is pulling it away. At the target, when P term is 0, if the prop is still going fast, the D term is nonzero and causes it to brake. This way the prop will eventually stop at the target. The greater the D gain, the sooner it will happen. But be careful, too great D gain will cause the prop to go towards the target very slowly.

So, it seems like P and D terms can bring the prop to the target very well. Why do we need an I term then? Well, when we have gravity, the prop won't actually stop at the target, but a bit below it. Why? When the prop is floating without movement at the target, both P and D terms are 0, and yet the gravity is still pulling the prop down. The gravity force isn't countered by anything, so the prop will fall down... to the point where P term and gravity are in balance. You can make this effect smaller by increasing the P gain, but you will never get it exactly to the target this way.

The I term can fix this situation. When the prop floats below it's target, the error is nonzero, so the I term grows. It grows, and grows, and grows, to the point where error reaches 0. But this is exactly what we wanted, error 0 means we are exactly at the target

I term cutoff

The growing property of I term can be a trouble sometimes. If we start very far from our destination, the error is big and the I term grows very fast. It can become so big, that the prop will fly off into space before it can stabilize. Usually I term is used only to correct small errors that P and D alone can't fix - that's why I term is often calculated only when P term or D term are low enough, that is, it is "cut off" when P and D are too big.

It can be done for example like this:

Code:

```
PTerm = GainP * Error
DTerm = GainD * $Error / $T

if(PTerm:length() < 10 & DTerm:length() < 10) {
    ITerm += GainI * Error * $T
} else {
    ITerm = vec() #make I term equal to 0 when P and D terms are too big
}
```

Summary - perfect position stabilizing E2

Time to put all the knowledge from this tutorial together and make a perfect stabilizing E2.

Code:

```
@name Position Stabilizing
@inputs TargetPos:vector E:entity
@persist T [ITerm Error]:vector
@persist GainP GainI GainD

#TargetPos is the position we want to hold prop E at

runOnTick(1)

if(first()) {
    ITerm = vec()
    #those are the values that worked for me - adjust them, if they don't work for you
    GainP = 5
    GainD = 1
    GainI = 4
}

#this will let us calculate DeltaT ($T)
T = curtime()

#calculate error
Error = TargetPos - E:pos()

#calculate P and D terms
PTerm = GainP*Error
DTerm = GainD*$Error/$T

#use I cutoff
if(PTerm:length()<50 & DTerm:length()<10) {
    ITerm += GainI * Error * $T
} else {
    ITerm = vec()
```

I hope that now you understand how PID works and how to make your own self-stabilizing contraptions

*Last edited by Fizyk; 03-03-2010 at 10:25 AM. **Reason:** Changed highlight boxes to code boxes*

Share

ATHEIST

My programs: BIOS - Alcyone - Calculator - Notepad - Movie Player
My tutorials: applyTorque - Quaternions - PID controllers
Some other things I made: FT Chip - RK4 Solar System

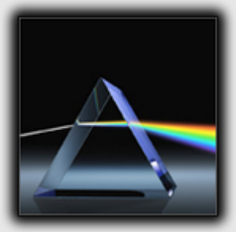
Reply With Quote

12-13-2009 #2

Matte

Developer

WIREMOD DEVELOPER



Join Date: Jan 2009
Location: Norway
Posts: 3,104

Re: PID controller - what is it and how to?

Good job, you really explained this well.

Share

"If anybody says he can think about quantum physics without getting giddy, that only shows he has not understood the first thing about them."
-- Niels Bohr

Reply With Quote

12-13-2009 #3

Whodunnit

Wire Sofaking

Whodunnit's Avatar
Join Date: Jan 2008
Location: New Zealand, Ackl
Posts: 636

Re: PID controller - what is it and how to?

Nice work, hopefully you will inspire more people to become physicists and engineers in the future

to read more about this stuff try control systems engineering textbooks.

Share

ЗАГРУЗКА.....

Reply With Quote

12-25-2009 #4

Dskodk

Wirererer

Re: PID controller - what is it and how to?

WIREMOD MEMBER

Join Date: Sep 2009
Location: In a toilet somewhere
Posts: 195

ooh, so this is physics... i like it... IMA BECOMING A PHYSICIST~!

Share

What do you call a man with no shins? Tony.

Reply With Quote

12-25-2009 #5

Maxaxle

Wirererer

Maxaxle's Avatar
Join Date: Sep 2009
Posts: 177

Re: PID controller - what is it and how to?

> Originally Posted by **Dskodk**
> *ooh, so this is physics... i like it... IMA BECOMING A PHYSICIST~!*

Quick. Get a crowbar.

Share

Okei likes this.

Still very clueless about code in general.

> *92% of teens have moved onto rap. If you are part of the 8% that still listen to real music, copy and paste this into your signature.*

Reply With Quote

12-26-2009 #6

Whosdr

Hickley Likes This

Join Date: Dec 2008
Posts: 2,300

Re: PID controller - what is it and how to?

Lol.

Share

.siht daer ot gniyrt emit detsaw ev'uoY

Reply With Quote

12-26-2009

Fizyk

Wire Sofaking

WIREMOD MEMBER

Join Date: Jun 2008
Location: Łomianki, Poland
Posts: 744
Blog Entries: 1

Re: PID controller - what is it and how to?

Matte, Whodunnit: thanks

Dskodk: yeah, this is one of the things you do when you are learning physics. Or rather, physics tells you more about why this works, it's more like engineering to learn how to make things like this

Share

ATHEIST

My programs: BIOS - Alcyone - Calculator - Notepad - Movie Player
My tutorials: applyTorque - Quaternions - PID controllers
Some other things I made: FT Chip - RK4 Solar System

Reply With Quote

12-26-2009 #8

Ehmmett

Wire Sofaking

Ehmmett's Avatar

Join Date: Sep 2009
Posts: 595

Re: PID controller - what is it and how to?

Alright so I tried using this. It doesn't stabilize angles and when I set things on it they just fall off
Am I doing it correctly?

Share

Reply With Quote

12-26-2009 #9

Fizyk

Wire Sofaking

WIREMOD MEMBER

Join Date: Jun 2008
Location: Łomianki, Poland
Posts: 744
Blog Entries: 1

Re: PID controller - what is it and how to?

Yeah, the example I presented here is only meant to stabilize the prop in position, which means it can still rotate freely. To stabilize angles too you need to add other PIDs, using quaternions, direction vectors or angles, whichever one you prefer.

For quaternions and direction vectors, your error will usually be the rotationVector() (or equivalent, like cross product, for direction vectors). For angles, the error will be the difference in angles.

Share

My programs: BIOS - Alcyone - Calculator - Notepad - Movie Player
My tutorials: applyTorque - Quaternions - PID controllers
Some other things I made: FT Chip - RK4 Solar System

Reply With Quote

12-26-2009 #10

krazyfool

Wire Amateur

krazyfool's Avatar
Join Date: Jul 2009
Posts: 77

Re: PID controller - what is it and how to?

Thanks man have a in fact have a whole batch.

Share

Reply With Quote

Page 1 of 2 **1** 2 Last

« Previous Thread | Next Thread »

Similar Threads

[Cam Controller] ColorMod on Cam Controller By emspike in forum Ideas & Suggestions	Replies: 8 Last Post: 10-08-2009, 02:08 PM
adv. pod controller bug By frog in forum Bug Reports	Replies: 1 Last Post: 08-18-2008, 01:49 PM
Turret with Adv. Pod Controller and Cam Controller? By Jaded Misanthrope in forum Installation and Malfunctions Support	Replies: 6 Last Post: 07-21-2008, 10:47 AM
Pod Controller By uoykcuf in forum Bug Reports Archive	Replies: 7 Last Post: 12-17-2007, 08:22 AM
How do I use the pod controller? By Tom in forum Installation and Malfunctions Support	Replies: 2 Last Post: 07-20-2007, 09:29 AM

Bookmarks

Digg
del.icio.us
StumbleUpon
Google
Facebook

Posting Permissions

You may not post new threads
You may not post replies
You may not post attachments
You may not edit your posts

Pingbacks are On
Refbacks are On

BB code is On
Smilies are On
[IMG] code is On
[VIDEO] code is On
HTML code is Off
Trackbacks are On

Forum Rules