Title:

Remove all jitter from your CH products with this script.

Text:

Some people including myself were experiencing small amount of jitter on their CH products, in my case, the CH pro throttle’s z-axis. For many, the easiest fix was to use a powered hub or plug the device directly into the motherboard for cleanest power supply. For me, neither worked and I’m sick of this jitter so I attempted to write a script to filter it. This is a pretty boring weekend and since ED apparently released the wrong patch, I decided to make this thread to help some of you out.

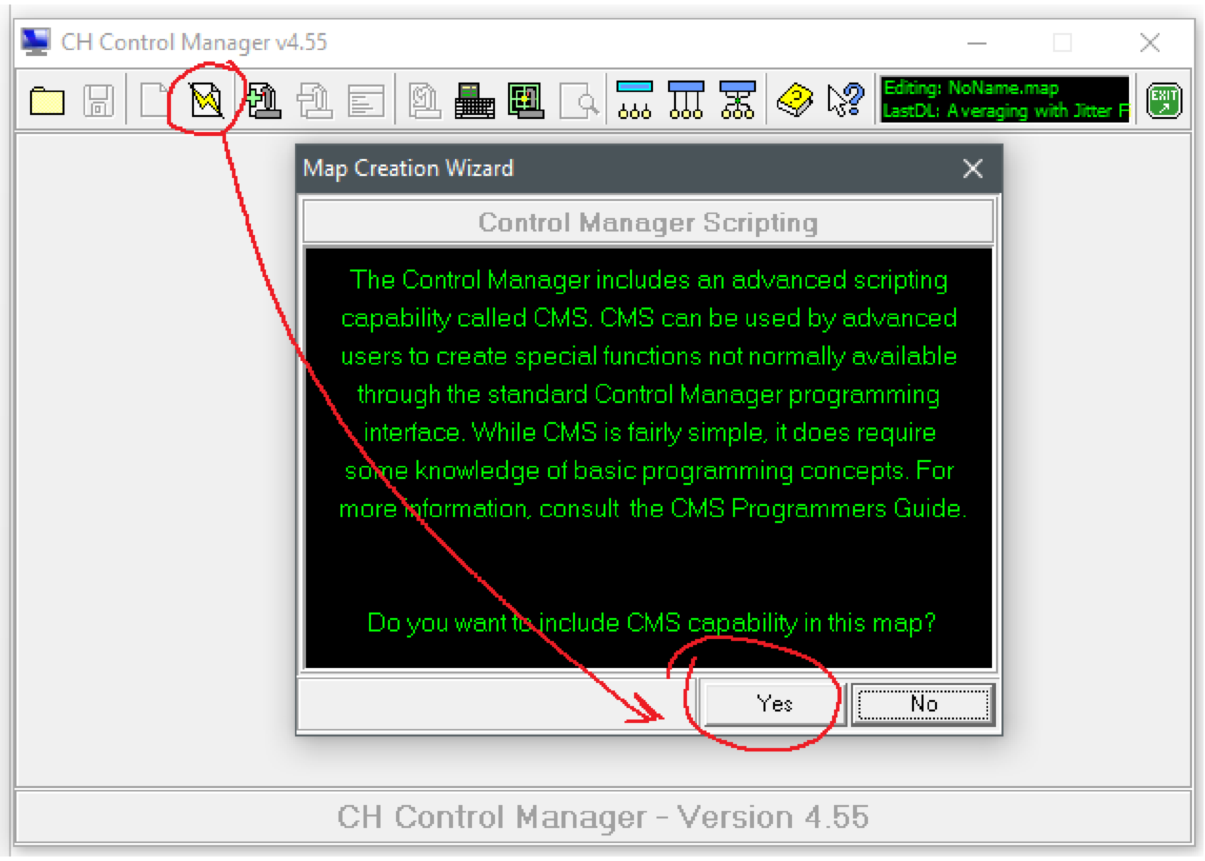
At first, a simple deadzone of +/- 1 worked but reduced the precision of the equipment by half. However, I noticed that because CH throttle uses a range of about 220 points and then during calibration it is scaled to 255, at some places there will be jumps of 2 on the axis so the jitter will also be +/- 2 so you need an even larger deadzone to compensate for this.

This is a terrible solution so I tried another way to filter out this jitter, and here are the steps to do it, I shall try and make it as noob friendly as possible. The easiest would be to simply load my map, but this may not work if you have multiple devices. Anyways, if your setup is exactly the same as mine, here is the file: https://drive.google.com/open?id=0Bxolnix8ukgRU0tNWlZSS2gzWjg

Step 1. Open CH control manager and calibrate your devices.

Step 2. Go through the control wizard and select the devices you are going to use for this map

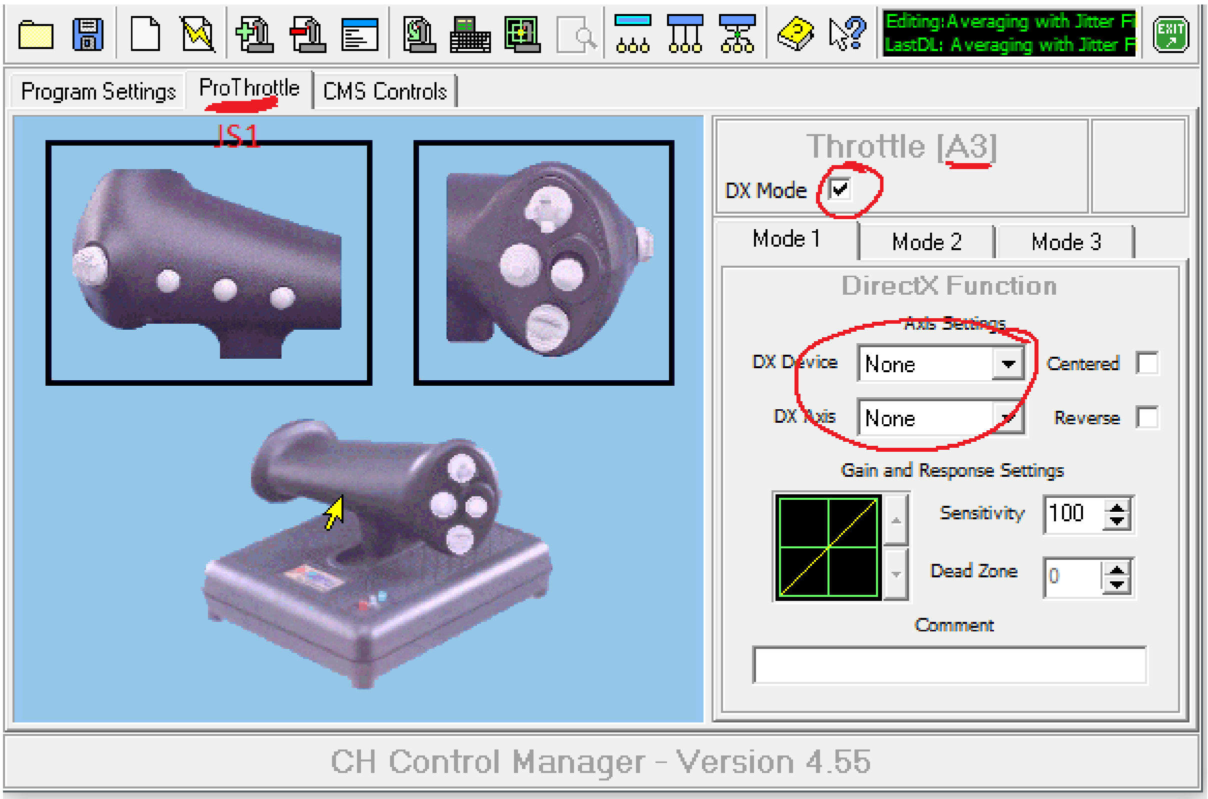
Make sure you include the CMS capability.



The Control Manager includes an advanced scripting capability called CMS. CMS can be used by advanced users to create special functions not normally available through the standard Control Manager programming interface. While CMS is fairly simple, it does require some knowledge of basic programming concepts. For more information, consult the CMS Programmers Guide.

Do you want to include CMS capability in this map?

Step 3. Select the axis you are experiencing jitter with, and set its DX Device and DX Axis to “None”.



Editing: Averaging with Jitter F
LastDL: Averaging with Jitter F

Throttle [A3]

DX Mode ☑

DX Device None   Centered □

DX Axis None   Reverse □

Sensitivity 100

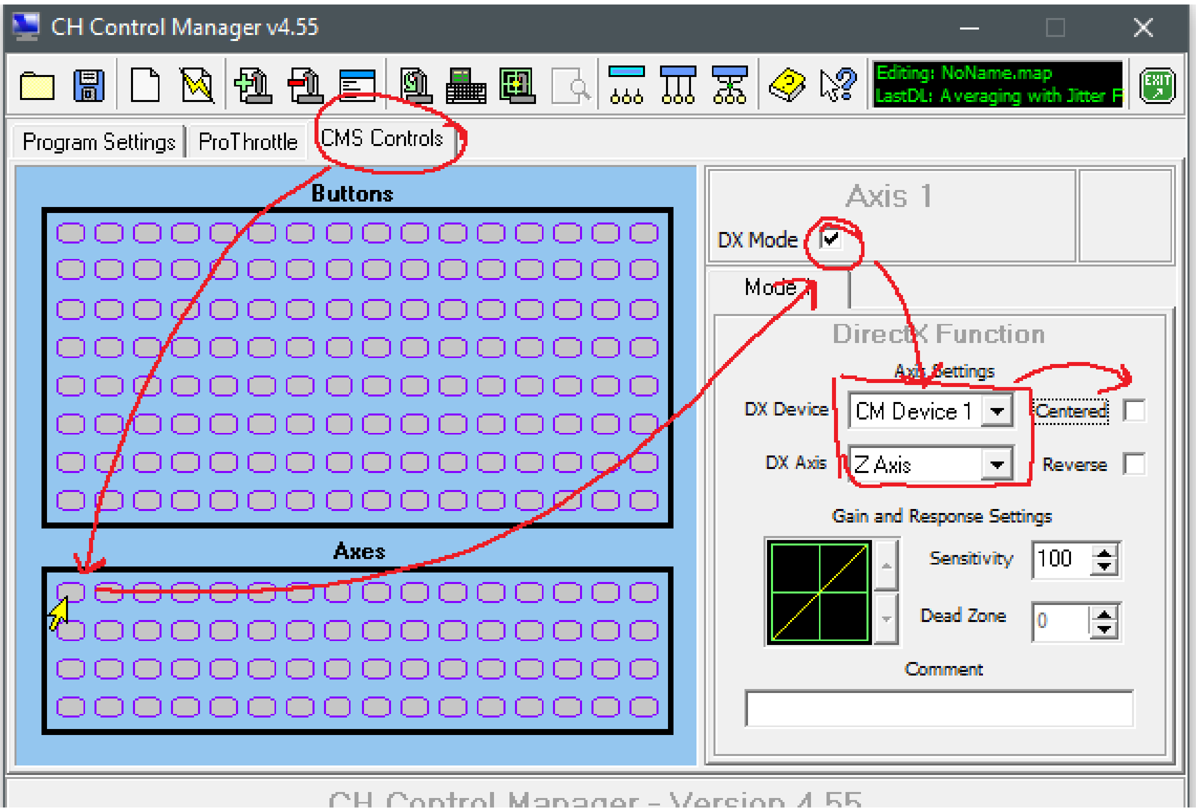
Dead Zone 0

CH Control Manager - Version 4.55

Note the underlined words, ProThrottle is my joystick 1 (JS1) and if you have more than one device, JS2 will be the device right of whatever is before it. Also note the “A3” underlined, that means the throttle axis ID is “a3”, this will be needed for the script.

Step 4.

Go the CMS Controls tab and select the axis 1 (or any other one). Change DX Device and DX Axis to the desired values, for me, I will use it as CM Device 1 and Z axis. *Make sure the “Centered” box is checked depending on the type of axis you use.*



Editing: NoName.map
LastDL: Averaging with Jitter F

Axis 1

DX Mode ☑

DX Device CM Device 1   Centered □

DX Axis Z Axis   Reverse □

Sensitivity 100

Dead Zone 0

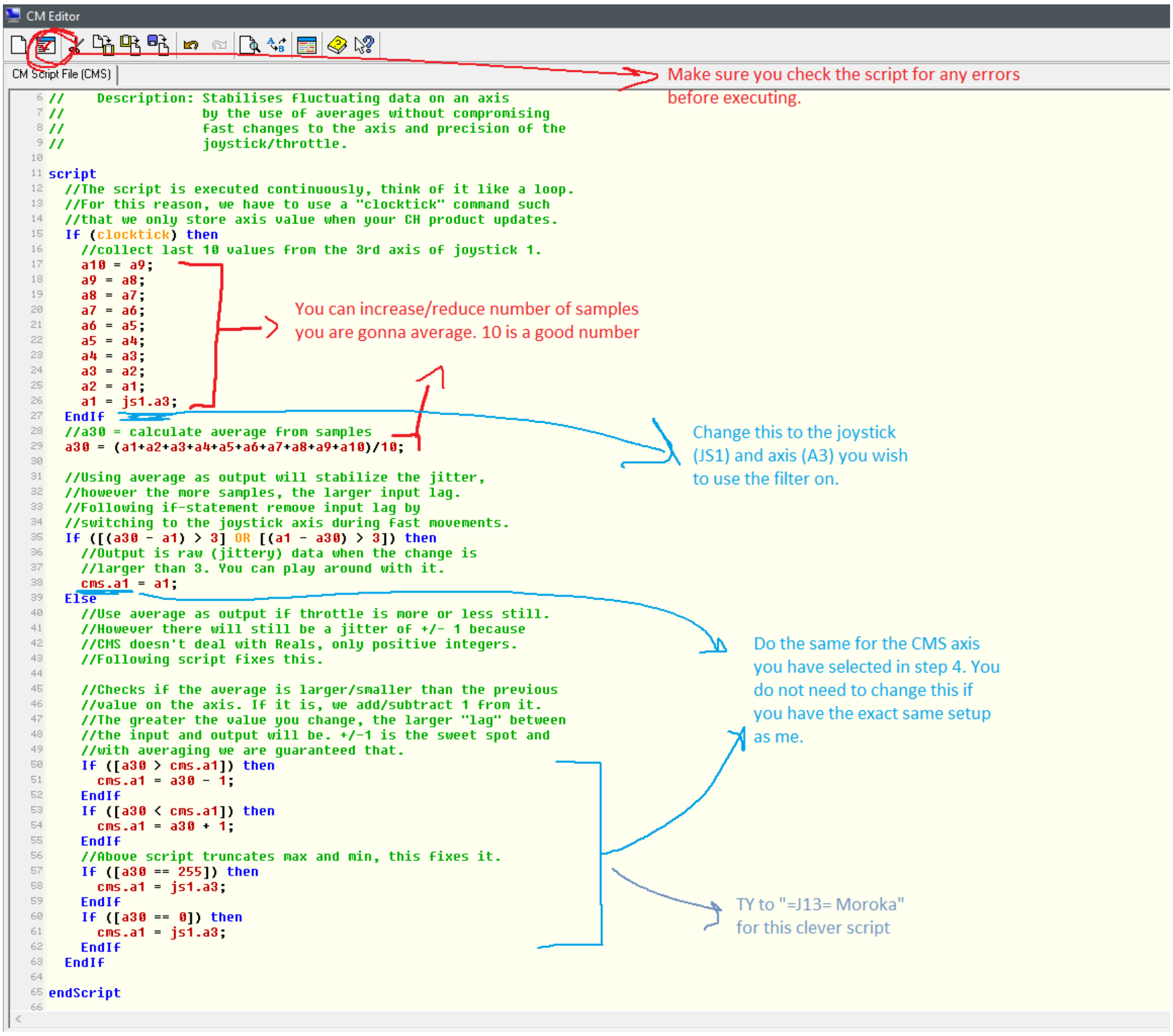
CH Control Manager - Version 4.55

Step 5. The script.

Open the CM Editor and paste in the following script:

```
// CMS Script File
//
//    Title: Jitter Reduction Script
//    Written By: Pure_Awesomeness
//          Date: 03/08/2016
//   Description: Stabilises fluctuating data on an axis
//               by the use of averages without compromising
//               fast changes to the axis and precision of the
//               joystick/throttle.

script
 //The script is executed continuously, think of it like a loop.
 //For this reason, we have to use a "clocktick" command such
 //that we only store axis value when your CH product updates.
 If (clocktick) then
  //collect last 10 values from the 3rd axis of joystick 1.
  a10 = a9;
  a9 = a8;
  a8 = a7;
  a7 = a6;
  a6 = a5;
  a5 = a4;
  a4 = a3;
  a3 = a2;
  a2 = a1;
  a1 = js1.a3;
 EndIf
 //a30 = calculate average from samples
 a30 = (a1+a2+a3+a4+a5+a6+a7+a8+a9+a10)/10;

 //Using average as output will stabilize the jitter,
 //however the more samples, the larger input lag.
 //Following if-statement remove input lag by
 //switching to the joystick axis during fast movements.
 If ([(a30 - a1) > 3] OR [(a1 - a30) > 3]) then
  //Output is raw (jittery) data when the change is
  //larger than 3. You can play around with it.
  cms.a1 = a1;
 Else
  //Use average as output if throttle is more or less still.
  //However there will still be a jitter of +/- 1 because
  //CMS doesn’t deal with Reals, only positive integers.
  //Following script fixes this.

  //Checks if the average is larger/smaller than the previous
  //value on the axis. If it is, we add/subtract 1 from it.
  //The greater the value you change, the larger "lag" between
  //the input and output will be. +/-1 is the sweet spot and
  //with averaging we are guaranteed that.
  If ([a30 > cms.a1]) then
   cms.a1 = a30 - 1;
  EndIf
  If ([a30 < cms.a1]) then
   cms.a1 = a30 + 1;
  EndIf
  //Above script truncates max and min, this fixes it.
  If ([a30 == 255]) then
```

```
      cms.a1 = js1.a3;
    EndIf
    If ([a30 == 0]) then
      cms.a1 = js1.a3;
    EndIf
  EndIf

endScript
```



```
//    Description: Stabilises fluctuating data on an axis
//                 by the use of averages without compromising
//                 fast changes to the axis and precision of the
//                 joystick/throttle.

script
  //The script is executed continuously, think of it like a loop.
  //For this reason, we have to use a "clocktick" command such
  //that we only store axis value when your CH product updates.
  If (clocktick) then
    //collect last 10 values from the 3rd axis of joystick 1.
    a10 = a9;
    a9 = a8;
    a8 = a7;
    a7 = a6;
    a6 = a5;
    a5 = a4;
    a4 = a3;
    a3 = a2;
    a2 = a1;
    a1 = js1.a3;
  EndIf
  //a30 = calculate average from samples
  a30 = (a1+a2+a3+a4+a5+a6+a7+a8+a9+a10)/10;

  //Using average as output will stabilize the jitter,
  //however the more samples, the larger input lag.
  //Following if-statement remove input lag by
  //switching to the joystick axis during fast movements.
  If ([(a30 - a1) > 3] OR [(a1 - a30) > 3]) then
    //Output is raw (jittery) data when the change is
    //larger than 3. You can play around with it.
    cms.a1 = a1;
  Else
    //Use average as output if throttle is more or less still.
    //However there will still be a jitter of +/- 1 because
    //CMS doesn't deal with Reals, only positive integers.
    //Following script fixes this.

    //Checks if the average is larger/smaller than the previous
    //value on the axis. If it is, we add/subtract 1 from it.
    //The greater the value you change, the larger "lag" between
    //the input and output will be. +/-1 is the sweet spot and
    //with averaging we are guaranteed that.
    If ([a30 > cms.a1]) then
      cms.a1 = a30 - 1;
    EndIf
    If ([a30 < cms.a1]) then
      cms.a1 = a30 + 1;
    EndIf
    //Above script truncates max and min, this fixes it.
    If ([a30 == 255]) then
      cms.a1 = js1.a3;
    EndIf
    If ([a30 == 0]) then
      cms.a1 = js1.a3;
    EndIf
  EndIf

endScript
```

Make sure you check the script for any errors before executing.

You can increase/reduce number of samples you are gonna average. 10 is a good number

Change this to the joystick (JS1) and axis (A3) you wish to use the filter on.

Do the same for the CMS axis you have selected in step 4. You do not need to change this if you have the exact same setup as me.

TY to "=J13= Moroka" for this clever script

Amend the joystick and axis if required (marked in blue).

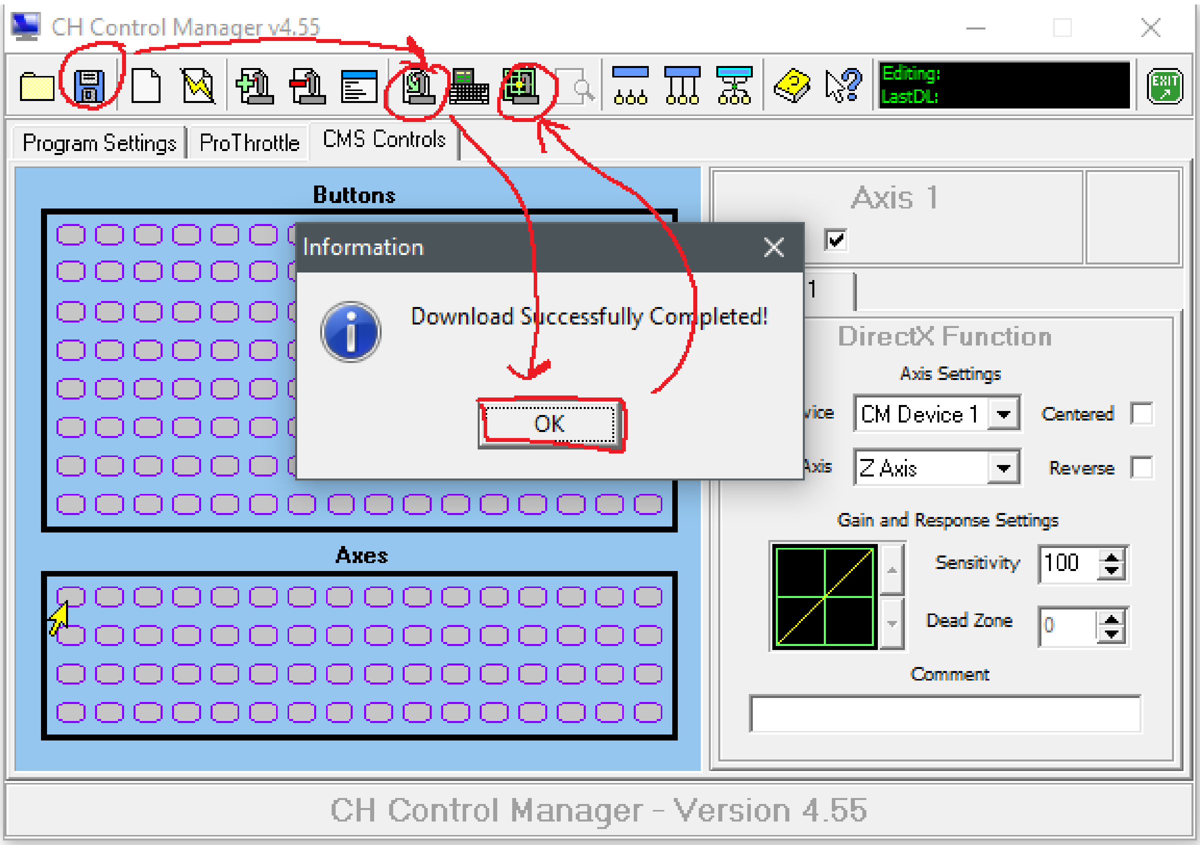
Feel free to make any changes to the script and improve on it, if you have made a breakthrough contact me, I will gladly update this document. I added a lot of comments so even if you are not familiar to programming you can grasp how the script works.

I used a small part of the script from a clever individual named “=J13=Moroka” in the following thread: http://forum.il2sturmovik.ru/topic/988-programmirovanie-dzhojstikov-ch-products/page-9#entry207047

This part completely eliminates +/-1 jitter but it only works if the jitter is exactly +/-1 and with larger values the virtual axis will lag behind the real one. I was quite lucky to stumble upon this piece of code.

Step 6.

Now you need to save and "download" the map we have made. Check that it works correctly by clicking on the "Test/Calibrate" button. You may notice that there is some initial jitter when you move the stick too quickly and then it settles down. This is because it takes a bit of time to take ten samples. You can reduce the number of samples to 5 or 3 but the average will be less stable, like I said, play around with it and use the numbers that work for you.



Information

Download Successfully Completed!