PID Tuning Guide (written by silverxxx)

This text is taken from here

Ok , the pids look something like this in file pid.c (rate) and anglepid.c (for level)

```
Quote:
// Kp ROLL PITCH YAW
float pidkp[PIDNUMBER] = { 17.0e-2 , 17.0e-2 , 10e-1 };
// Ki ROLL PITCH YAW
float pidki[PIDNUMBER] = { 15e-1 , 15e-1 , 50e-1 };
// Kd ROLL PITCH YAW
float pidkd[PIDNUMBER] = { 6.8e-1 , 6.8e-1 , 5.0e-1 };
// output limit
const float outlimit[PIDNUMBER] = { 0.8 , 0.8 , 0.4 };
// limit of integral term (abs)
const float integrallimit[PIDNUMBER] = { 0.8 , 0.8 , 0.4 };

in the brackets, the first number is roll, second pitch, and last is yaw.
```

Generally the roll and pitch are the same for square frames.

So you have kp , ki and kd which are the 3 terms of the pid.

Then there is outlimit, that sets the max pid output. In this case the output is at 0.8 , or 80% thrust max for pitch and roll, and 40% for yaw (because more yaw makes it unstable)

The integrallimit is similar, but applies to the Ki Iterm only. It can be set the same as outlimit , or smaller.

About the numbers used: because this are float numbers, and there are radians somewhere in the program, the p, i , and d terms are very small, so they use the scientific notation, in order to not have to type many zeros.

So 17e-2 = 17 * 10 to the power(-2) = 0.17

and 15e-1 = 1.5

There are also large positive numbers in some places, for instance 1e6 is 1000000

Level mode tuning:

In level mode, the firmware uses both rate and angle pid, and as such, it's best to have the quad tuned in acro mode before fine-tuning level mode. The level mode pid settings are found in file "*anglepid.c*"

Level mode is pretty easy to tune as it usually only needs P or PI (it will fly ok with P only). For quick tuning set P only, and set I to zero.

Acro mode tuning:

It's possible to have the quad fly with P and D only, and that is probably the fastest way to tune it

since only 2 numbers are required. From PD to PID generally the P term can be reduced about 30%-50% and I term increased to a value where there are no side effects.

I'd start with D of 5e-1 and find a P that works well. If the quad flies, it's easy to increase D, as a d that's too high can generally either be heard, or induce very fast oscillations. If the quad is drifty (and hard to control) it needs more P, if it oscillates after a change of angle , it needs less.

A P that is too high will cause oscillations after a stick input. In this case, either decrease P or increase D.

The I term can have all sorts of effects, and it is also affected by P (and vice-versa) so it's best to either leave it last, or have a small one (or zero) The I term can cause some overshoot, or it can cause drift if its zero and P and D are not high enough. It can also cause oscillations that get worse with time. In case of overshoot, lower I term or increase P term.

Note that a quad with P D (i set zero) will only fly well if the motors produce similar thrusts, and the quad is built straight. The CG has to be in the middle, as well.

The D term usually creates the “locked in” feel. The higher the D term, the more “locked in” the quad will feel, and also will cope better with wind and other issues. A higher D will also allow a higher P to be used, hence increasing control response. Too high D will cause very fast oscillations, and it may also cause “jumpyness”, the quad seemingly jumping up/down. The jumpiness is caused by vibrations (mainly from the motors), as the D term is vibration sensitive.

A unusual method to check how good the quad is tuned is to hit it while hovering with an object (ruler / pencil) slightly, on one of the corners. If it oscillates afterwards, either/both P or I are too big or D is too small.

$$i^{-n} = \frac{1}{a^n}$$