Wind-Adaptive Self-balanced control for Quad-copter using embedded method

Undergraduate:

HUANG ZILONG, YI WEIYING

Supervisor:

Professor Leung Hong Fung

Professor. Mak Sui Tung

1. Introduction:

Quad-copter, also known as quad-rotor, has recently become the most welcomed UAV (Unmanned Aerial Vehicle) among research team all over the world because of its VTOL (Vertical Take Off & Landing) feature, simplicity in flying mechanism**[i]** and small size.

However, miniature quad-copter in flight is influenced badly by sudden airflow, which will cause serious fluctuation, leading to deviation to the original course or even plane-crash.

To make a better chance for a quad-copter to survive in sudden airflow, we come up with an idea: Why not give some adaptability to the quad-copter, so that it can switch between 'windy mode' and 'normal mode', to increase the stability in windy condition[1] and also have good performance in normal condition[2].

That is, to change the P, I, D values of its PID controller (See 2.2) to attain more stability in windy condition rather than using the original PID values that achieve stability in normal condition.

Nevertheless, the experiment result does not sustain our idea of changing PID parameters, because neither a new set of PID brought overall better stability or did not improve the stability in much extent. Some analysis is done to find out the reason for this failure.

---

[1] Windy Condition means that there exists wind in the environment and the deviation of the roll, pitch and yaw angle could be very large with respect to the deviation in normal condition.

[2] Normal Condition means indoor, or outdoor with mild wind.

2. Theory

2.1 NED (also called local tangent plane) coordinate system

For an UAV, we must set up a coordinate system at its COG (center of gravity) in order to study its movements, attitude and stability. The NED coordinate system **[ii]** is a very common choice (as illustrated by figure 1).

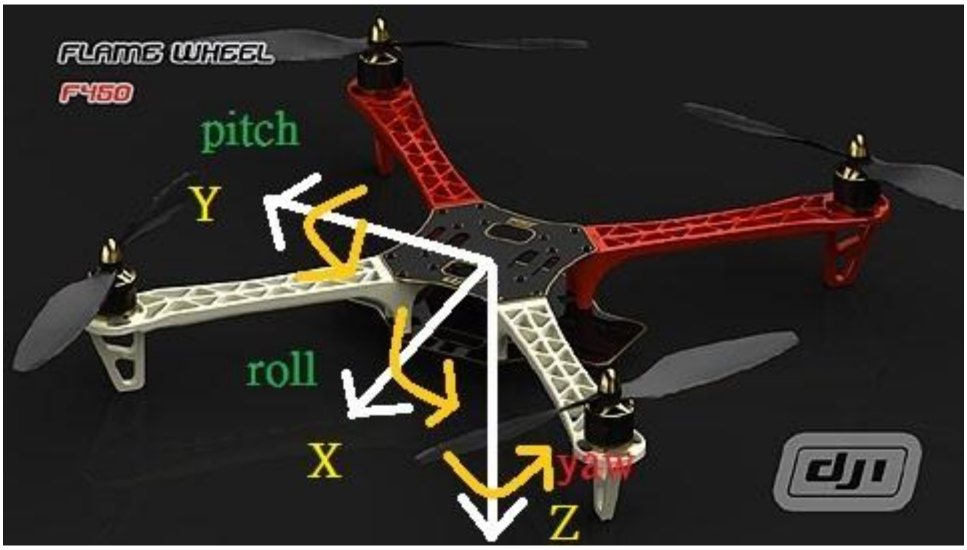


Figure 1: The NED coordinate system

The X axis is pointing forward, while Y axis pointing rightward and Z axis pointing downwards. Roll (φ), pitch (θ) and yaw (ψ) are the rotations (in angle) around X, Y, Z axis. Therefore, a quad-copter's attitude can be fully explained by roll, pitch, yaw these three parameters.

And one can see that the stability of a quad-copter is mostly related with the roll and pitch angle, which tell you whether the quad-copter is holding its balance horizontally. The yaw angle doesn't really affect the stability.

2.2 PID controller used by Paparazzi open-source project [iii]

2.2.1 PID Controller

WITH its three-term functionality covering treatment to both transient and steady-state responses, proportional-integral-derivative (PID) control offers the simplest and yet most efficient solution to many real-world control problems.

- *The proportional term—providing an overall control action proportional to the error signal through the all-pass gain factor.*
- *The integral term—reducing steady-state errors through low-frequency compensation by an integrator.*

- *The derivative term—improving transient response through high-frequency compensation by a differentiator.* **[iv]**

Because a PID controller does not require knowledge to the physical model, so it's widely used in control system of quad-copter, which can save computing power (miniature quad-copters are usually equipped with MCU with low frequency and small power consumption to increase the flight time) and is more robust (more adaptive to complicated environment).

2.2.2 Control System of Paparazzi Open-Source Project[v]

We're using Paparazzi Open-Source project's material to make our quad-copter, so the modification is based on their algorithm. The Paparazzi Open Source Project is using a variant of PID controller, adding a feed-forward term together with upper Bound of the Integral Term to improve the performance.

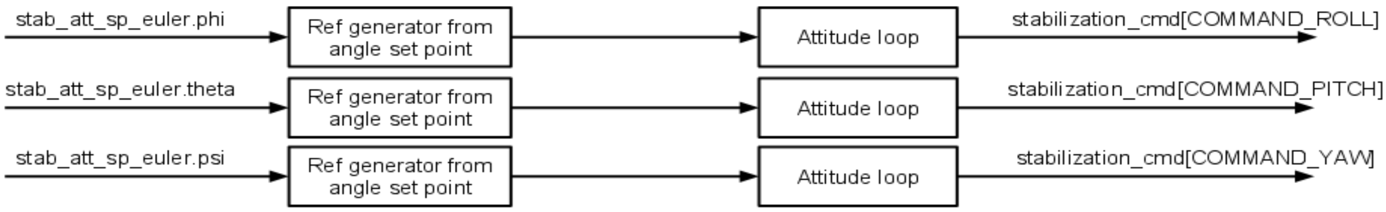


Figure 2: Stabilization Control loop Overview of Paparazzi Open Source Project

The stabilization Control Loop takes the set-points of three angles – roll (phi), pitch (theta), and yaw (psi) as inputs and finally computes the corresponding commands. For details of Ref generator and Attitude Loop, see figure 3 and figure 4.

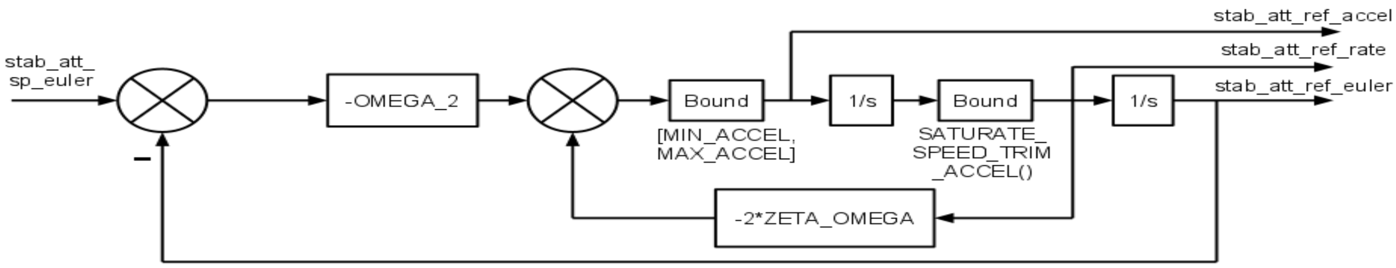


Figure 3: Details of Reference Generator

The set-point input is used to computed the set-point value for acceleration, angular speed and Euler angle

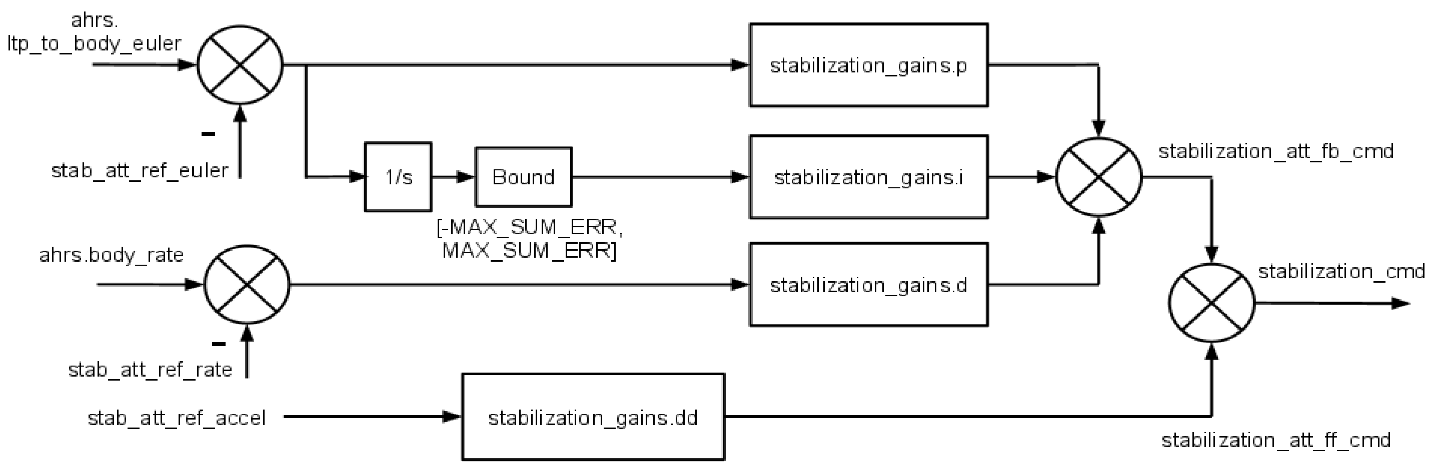


Figure 4: Details of PID Attitude Loop

Remark: Figure 2 -4 are from http://paparazzi.enac.fr/wiki/Control_Loops#Horizontal_Control

This is the core part using PID controller for the attitude stabilization. The controller can be represented in the following formula.

1. $error_{angle}(t) = estimation_{angle}(t) - ref_{angle}(t)$
2. $error_{rate}(t) = estimation_{rate}(t) - ref_{rate}(t)$
3. $stabilization_{fbcmd}(t) = error_{angle}(t) * P + \sum_{\tau=0}^{\tau=t} error_{angle}(\tau) * I + error_{rate}(t) * D$
4. $stabilization_{ffcmd}(t) = radioControl * dd$
5. $stabilization_{cmd}(t) = stabilization_{fbcmd}(t) + stabilization_{ffcmd}(t)$

2.3 Simulation and Proof of Feasibility

Usually for a PID controller, it's very hard to achieve short transient time and high stability at the same time (and the simulation using Matlab proves that). A quad-copter in normal condition will not suffer from sudden change of the attitude as the influence of external force, so its stability is more emphasized, and the transient time[3] is not required to be short.

To simulate a PID controller, we defined the following:

i. $est(t+1) = \frac{P}{kilo} * err(t) + \frac{I}{kilo} * \sum_0^t err(\tau) + \frac{D}{kilo} * \big(err(t) - err(t-1)\big) + est(t)$ where kilo = 1000
ii. $err(t) = setpoint - est(t)$
iii. est(t) is a simulation for roll or pitch or yaw angle, and it's an estimated value that comes from the IMU module in a real quad-copter.
iv. No disturbance exist in the system, that is, the est(t + 1) is strictly the output of PID controllers at time = t without any error.

[3] In this Matlab simulation, transient time is denoted as tt, and is defined as the time from t = 0 to the time when the est(t) has been bound by the threshold for a continuous period of time.

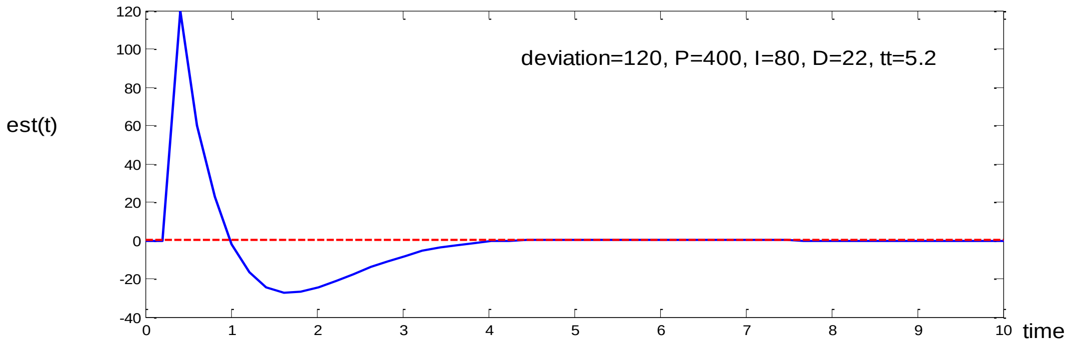


Figure 5: Matlab simulation of PID controller, with beginning deviation 120, P = 400, I = 80, D = 22, no noise, setpoint as 0

The Matlab simulation result of figure 5 gives out transient time as 5.2.

2.3.2 Transient time for longer deviation

For different deviations, the transient time also varies.(See figure 6)

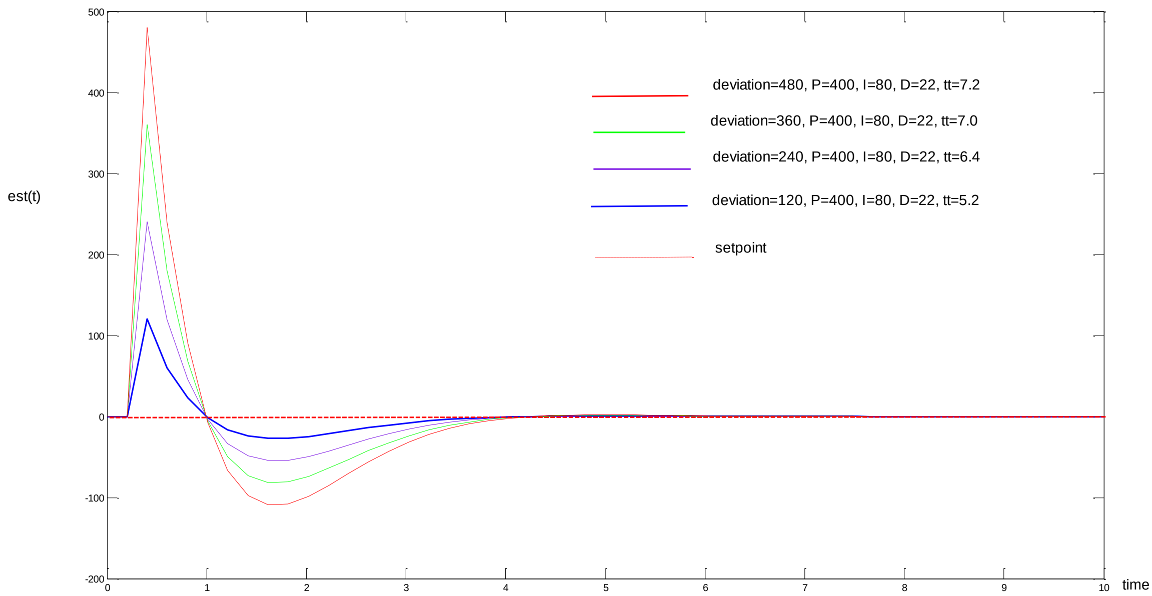


Figure 6: Matlab simulation for deviation as 480, 360, 240 and 120, PID values are the same

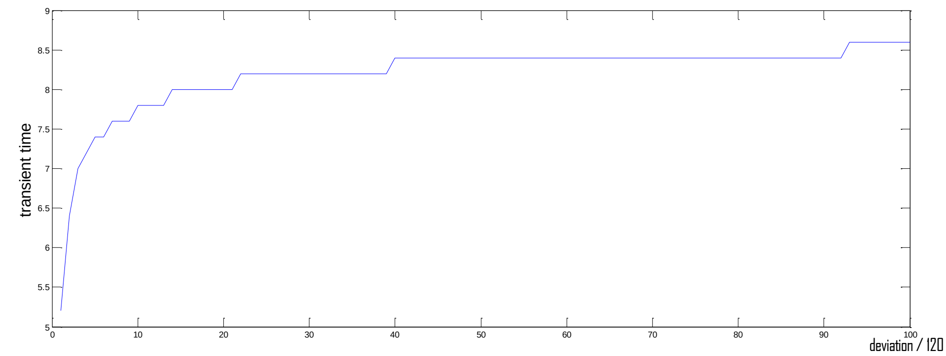


Figure 7: Matlab simulations for growing deviation versus transient time

Figure 7 reveals a fact that, the larger the deviation, the longer the transient time, but the transient time does not show a proportional relationship with the deviation.

The transient time grows in a logarithm manner, slowing down when approaching some maximum value. Yet, a larger deviation time will cause longer transient time which directly affects the safety of the quad-copter. The reason is that when the quad-copter need longer time to restore its attitude, it might end up with a high velocity and have already crashed into some obstacles or deviates far away from its pre-destined course.

2.3.3 Possible solution to shorten transient time for large deviation

Windy condition leads to large deviation, and long transient time we cannot tolerate, so it's needed to find a way to shorten the transient time under large deviation. (See figure 8)

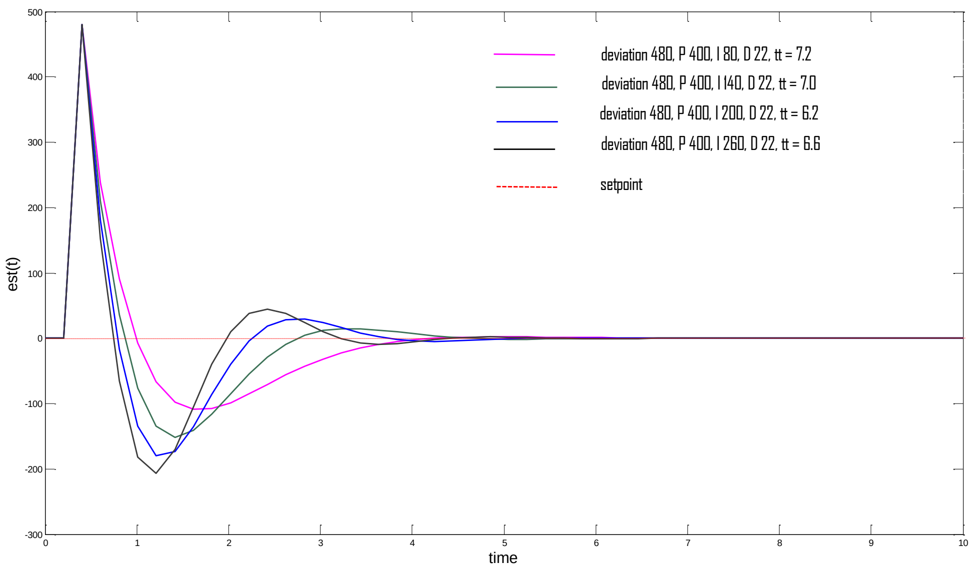


Figure 8: Relatively large deviation with different I values

The matlab simulation result given by figure 8 shows that by modifying the PID values, it's possible to shorten the transient time, but the modification should be careful because the transient time may bounce back as you over tuned those parameters.

2.3.4 Drawback to the proposed solution

However, a set of PID values that shorten transient time for larger deviation may prolong the transient time for small deviation. (See figure 9)

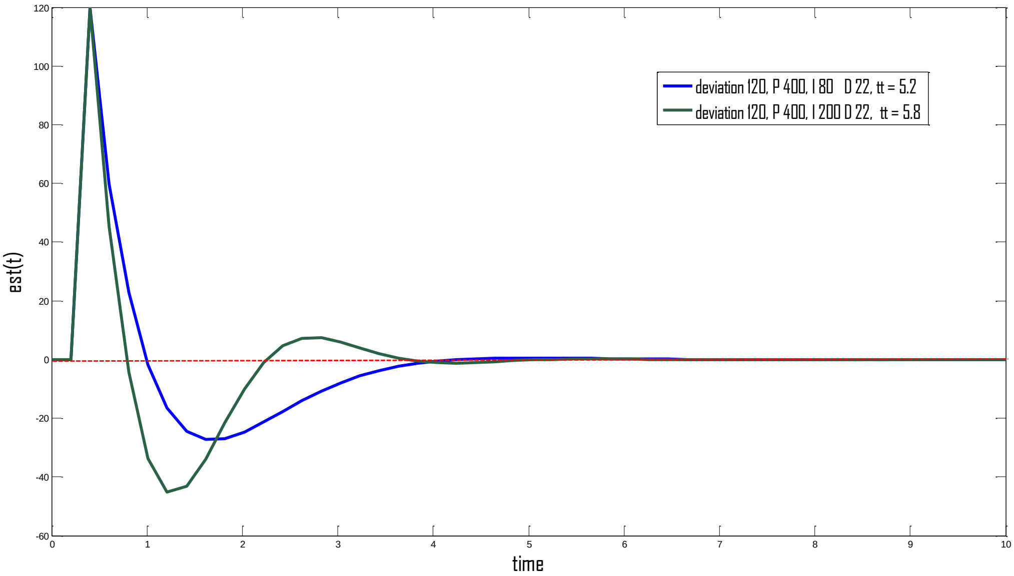


Figure 9: Small deviation with different I values.

The set of PID values (P 400, I 200, D 22) which provide shorter transient time when deviation is 480 gives out a longer transient time when the deviation is 120.

2.3.5 Simulation Conclusion:

The all above simulations give the following results:

1. **Large deviation will cause longer transient time for a set of PID values works well with small deviation.**
2. **Long transient time caused by large deviation can be reduced by properly setting PID values.**
3. **A set of PID values that provides better performance for larger deviation does not necessarily also gives good result for small deviation (usually bad results).**

<u>It's not practical to find an all-might PID values that can both works well in normal condition and windy condition.</u>

<u>Substitute the PID values in different condition will be a good solution.</u>

3.Method

3.1 General Idea

A set of PID values that works perfectly of the control systems in no-wind condition may have trouble dealing with windy condition, so changing the PID values can improve the performance of the control system to keep balance.

3.2 How to observe

The estimated roll, pitch and yaw angles are recorded during the flight. For a quad-copter with high stability, the estimation of the roll, pitch, yaw angle should stick closely to the set-point (usually 0 degree) of the roll, pitch and yaw angle.

To simplify our experiment, we have the following assumption(s) and definition(s).

Assumption I:

The quad-copter is a symmetrical rigid body. **[vi]**

Assumption II:

$PID_{Original}$ is already the optimal PID parameters in normal condition.

Definition I: Stability

If the deviation of roll (φ), pitch (θ) or roll (ψ) to their set-point is bounded in 2 °, we say the stability is achieved. **[vii]**

Definition II: $\boldsymbol{PID_{Origin}}$

$\boldsymbol{PID_{Origin}}$ is a set of PID values that provides optimal performance in normal condition.

Definition III: $\boldsymbol{PID_{Windy}}$

$\boldsymbol{PID_{Windy}}$ is a set of PID values that provides good performance in windy condition.

Remark I:

The deviation of yaw angle for a quad-copter is not taken into consideration.

(Remark I is based on the fact that even if a quad-copter is rotating about the Z-axis, it will not crash or drift from the original position.)

3.3 Wind Simulation Method

By studying the flight log, we found that wind will cause the est(t) (of roll, pitch or yaw)to deviates from its setpoint, usually in a random manner. So we have two ways to simulate windy condition.

Method 1: One way is to change the set-point, this has the same effect as sudden change of deviation, but it's just a very simple simulation of wind because the effect of winds is always varaible, in this kind of simulation, because the deviation changes in a pre-destined manner, so we can analyze the different transient time brought by different PID parameters.

Moreover, method 1 allows us to look into a single deviation, the basic component of continuous deviation in a real flight.

Method 2: The other is to use a fan to generate real wind on the quad-copter, and analyze the effect on stability by changing PID parameters. Because the deviation will come in a quite random manner, so you cannot simply separate one deviation from another deviation, the deviation will overlap and therefore transient time is complicated.

In order to have better and more convincing experiment result, we decide to use the first way to find the PID_Windy and then put it into test in the second way (which is much closer to a real windy condition).

3.4 Experiment Procedure

Part 1: Using wind-simulation method 1

1. ***Use wind-simulation method 1 to find a set of PID values(PID_{Windy}) that provides short transient time.***
2. ***Verify the simulation result about the effect on transient time caused by change of PID values***
3. ***Analyze the deviation during flight.***

Part 2: Using wind-simulation method 2

1. ***Analyse the deviation during flight of roll angle in normal condition with $PID_{Original}$.***
2. ***Analyse the deviation during flight of roll angle in windy condition with $PID_{Original}$***
3. ***Analyse the deviation during flight of roll angle in windy condition with PID_Windy.***
4. ***Compare the result from step 2 and step 3, to see whether the performance in windy condition is improved.***
5. ***Test PID_{Windy} in normal condition, to see whether it performs worse than $PID_{Original}$.***
6. ***Repeat step 1 – 5 for pitch angle.***
7. ***Analyse the data log in step 1 and step 2 to see how to characterize 'windy' condition for the quad-copter.***
8. ***Design a autonomous PID parameters changing system for the quad-copter.***

Remark: Part 2 step 7 and step 8 is not implemented because of our failure in the previous steps.

In order to constraint the deviation to only one axis (or say, angle) at a time, we made a steel/wood framework so that we can fix our quad-copter on it (see figure 10).

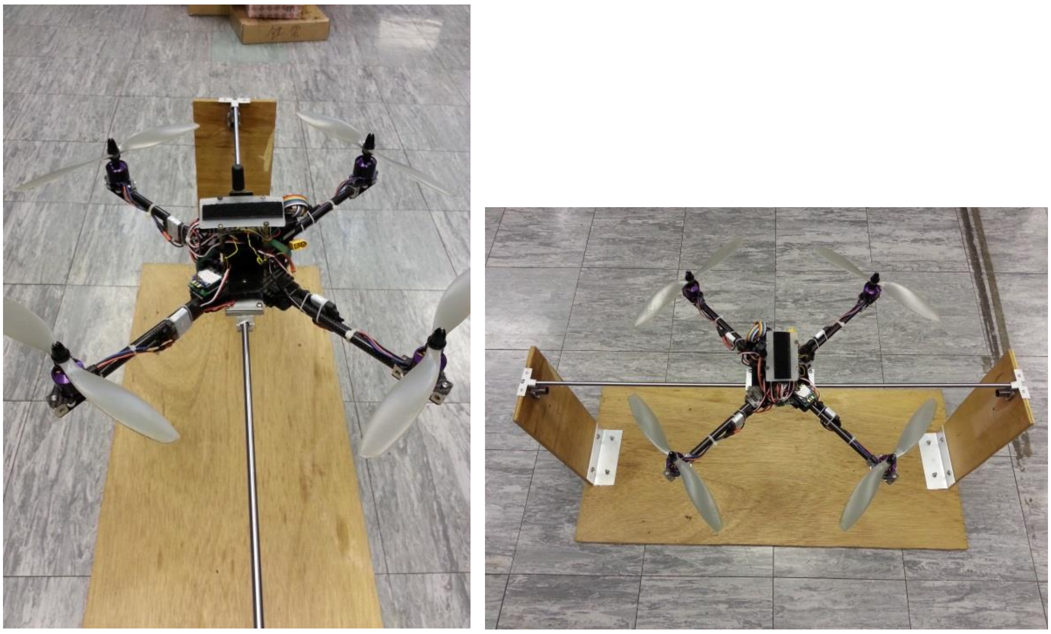


Figure 10: Steel/wood frame that fixes the quad-copter to only one degree of freedom

For wind simulation method 2, we have a fan continuously blowing on the frame.

Figure 11: The fan we use for windy condition simulation.

4. Experimental Result

Remarks: All of the following results are carried out by using a fully-charged 11.1V 3S Li-Po battery for 300s flight time fixed on the steel/wood frame. By doing this we can effectively eliminate the effect from unwanted variable (e.g. different power supply results in low rotational speed of the propellers, causing deviation).

4.1 Result from $PID_{Original}$ in Normal Condition

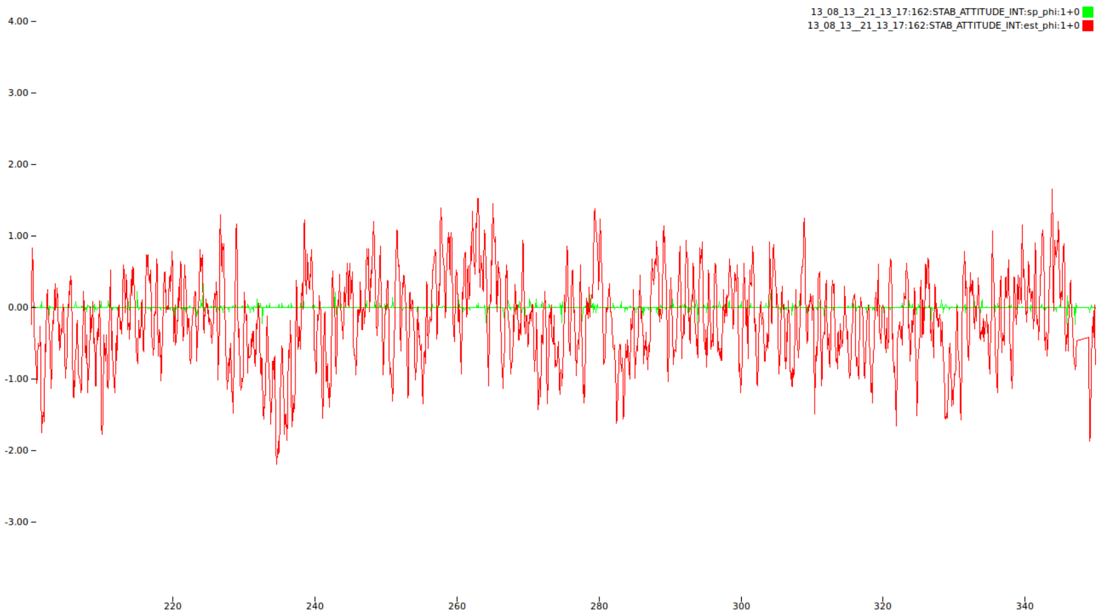


Figure 12: Roll angle estimation in normal condition with $PID_{Original}$.

X-axis of figure 12 is time and the Y-axis is the estimation of roll angle.

This figure shows how roll angle estimation deviates from the set-point 0. The maximum deviation reaches 2.2°, and the estimation values are mainly distributed around 0°. The flight lasted for about 150s, and the following results are also from a fixed-quad-copter flight for duration about 150s to eliminate the influence from the battery.

So the ***stability*** is achieved in normal condition with $PID_{Original}$.

4.2 Result from $PID_{Original}$ in Windy Condition (Wind simulation method 2)

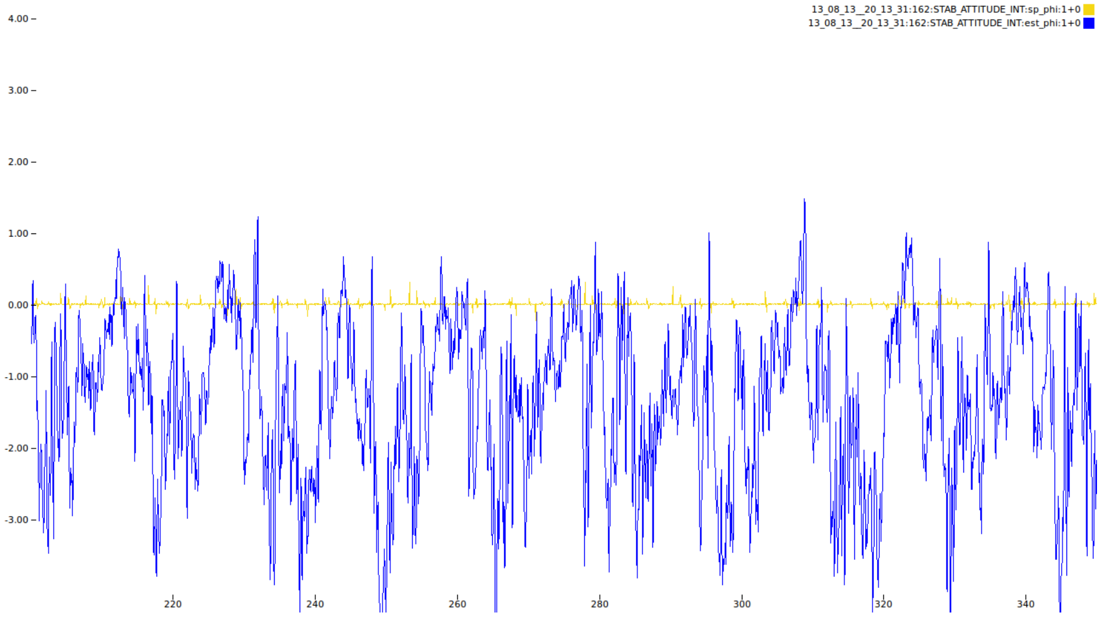


Figure 13: Roll angle estimation in windy condition with PID_Orginal

As usual, the x-axis is time and y-axis is the estimation of the roll angle. Compare figure 13 and figure 12 you can find that the roll angle is continuously reaching -4° and sometimes even get smaller than that. And the estimation is mostly distributed around -2°, all of those above indicates the quad-copter cannot achieve ***stability.***

So the windy condition does bring a lot of troubles to a quad-copter using PID_Original.

By comparing the figure 13 and figure 12, it can be said that the sudden deviation is usually larger (windy condition) in figure 13, and the reaction of the controller to the large deviation is too slow, and most of the time the deviation is negative. Combing the above analysis, we think the P parameter or the I parameter should be increased to shorten the transient time.

4.3 Result of changing PID values to have shorter transient time

Remark: The following experimental results are achieved in wind-simulation method 1.

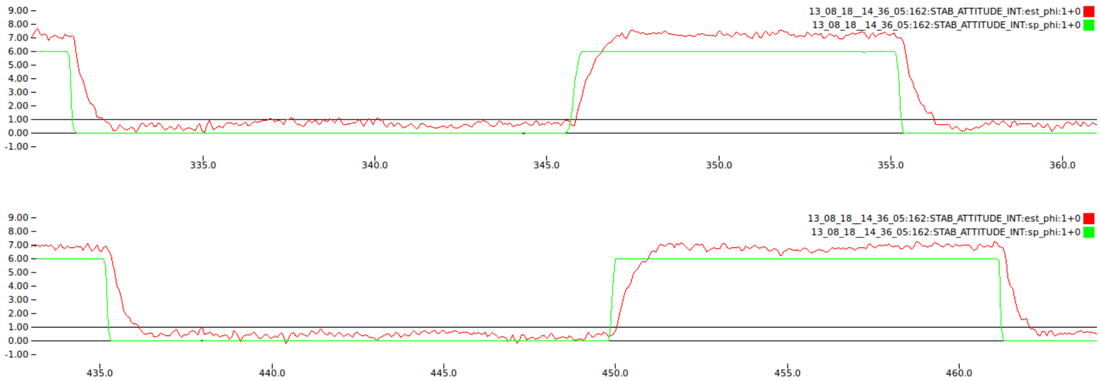


Figure 14: Roll angle estimation with different PID values after two sudden change of set-point

Figure 14 gives out the results of estimation of roll angle for a single deviation under different PID values.

As we scaled up the P parameter and I parameter, the transient time gets shorter, which did satisfy our requirement for windy condition. When the set-point stays at 0, the estimation of roll angle also stays inside the area bound by the two black lines, so it's quite stable.

But the transient time should be as short as possible, so the PID parameters were still scale up to meet the need.

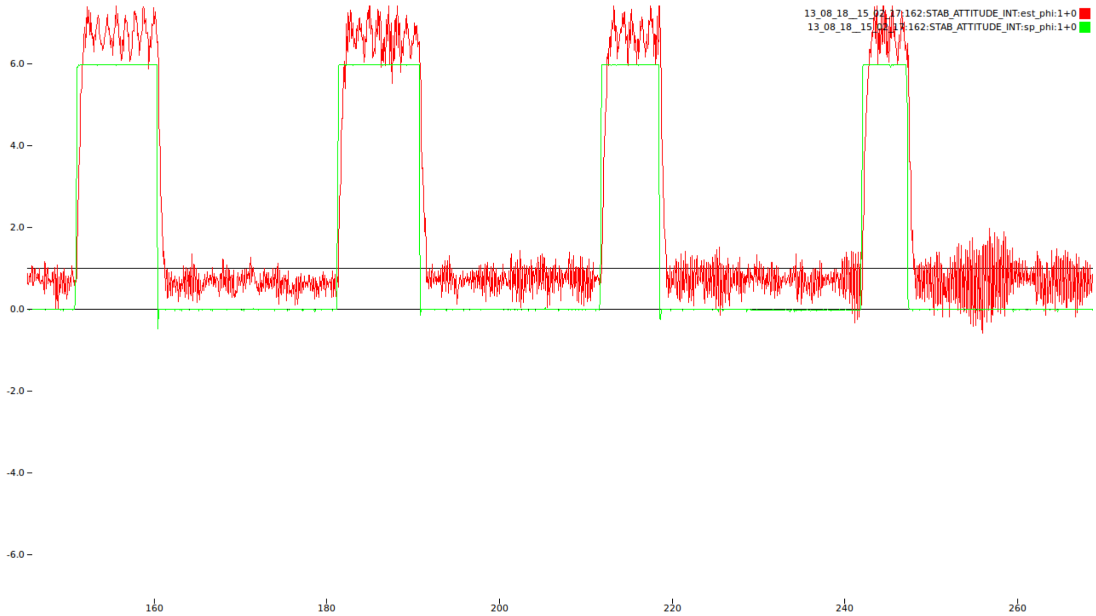


Figure 15: Roll angle estimation with even larger PID values after sudden changes of set-point

Figure 15 gives result of even larger PID values, the transient time is even shorter than before, but oscillation occurs, lucky enough the system is still stable.

4.4 Result from PID_{Windy} in Wind-simulation method 2

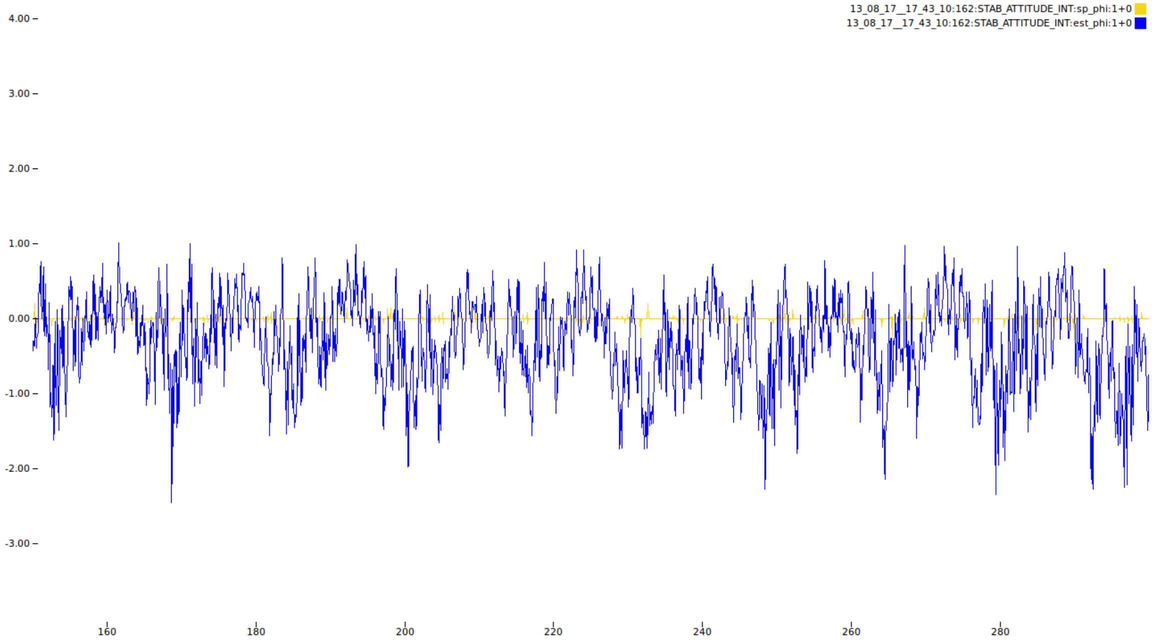


Figure 16: Roll angle estimation with PID_{windy1} having transient time tt_{windy1}

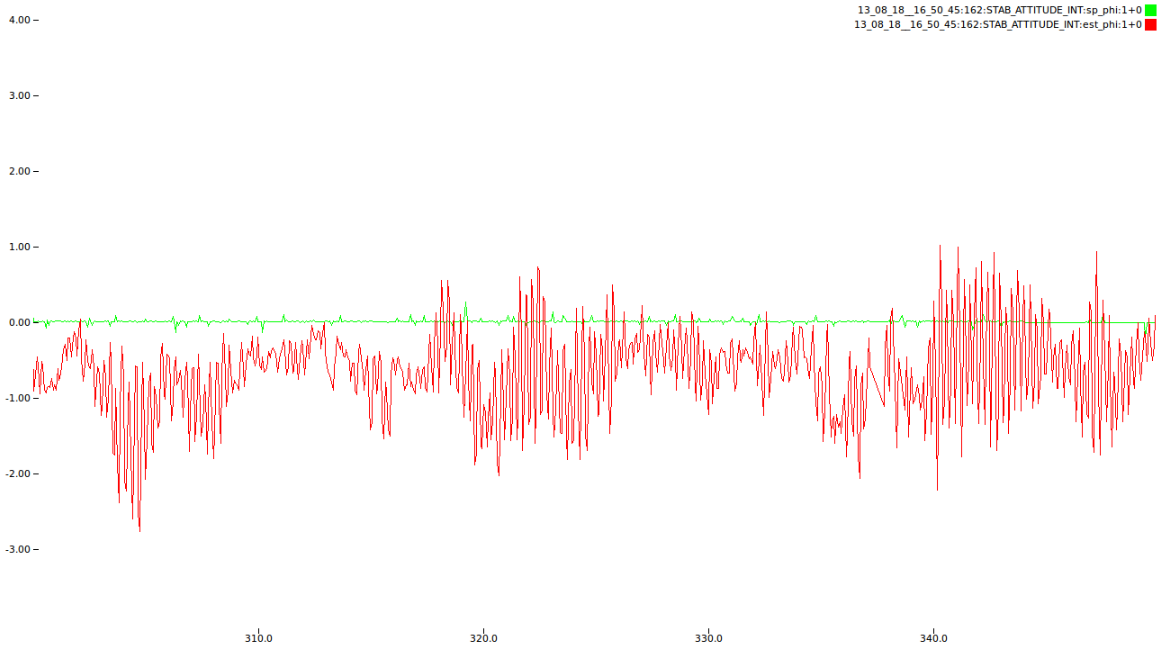


Figure 17: Roll angle estimation with PID_{windy2} having transient time tt_{windy2}

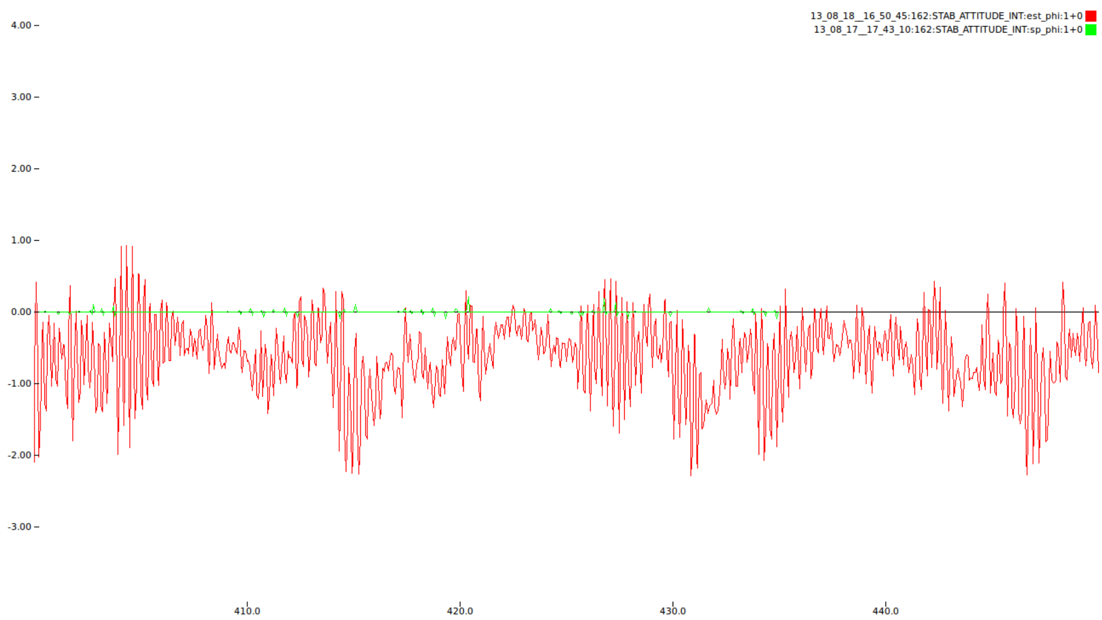


Figure 18: Roll angle estimation with PID_{windy3} having transient time tt_{windy3}

The ordinal relationship is:

$tt_{windy3} < tt_{windy2} < tt_{windy1}$

However, even though the transient time is shorter, PID_{windy3} didn't seem to give out a better result in wind-simulation method 2 compared to PID_{windy2} or PID_{windy1}. But all the PID_{windy} perform much better than $PID_{Original}$ in wind-simulation method 2.

4.4 Result from PID_{Windy} in normal condition

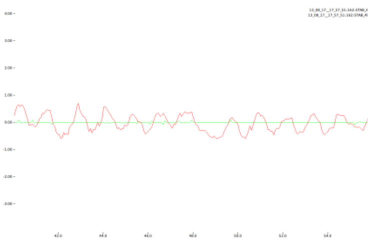


Figure 19: Roll angle estimation with PID_{windy1}

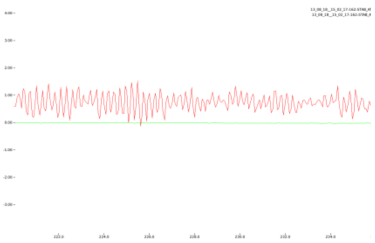


Figure 20: Roll angle estimation with PID_{windy2}

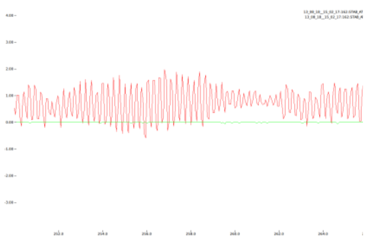


Figure 21: Roll angle estimation with PID_{windy3}

Remark: Large figure of 19, 20, 21 is below.

It's quite obvious that PID_{windy1} gives a much better performance than PID_{windy2} and PID_{windy3}, as PID_{windy2} and PID_{windy3} made the estimation of roll angle going back and forth so frequently in the same short period of time, despite the range is almost the same at 2°. The high frequency oscillation of the roll angle directly leads to lack of maneuverability and even hidden instability.

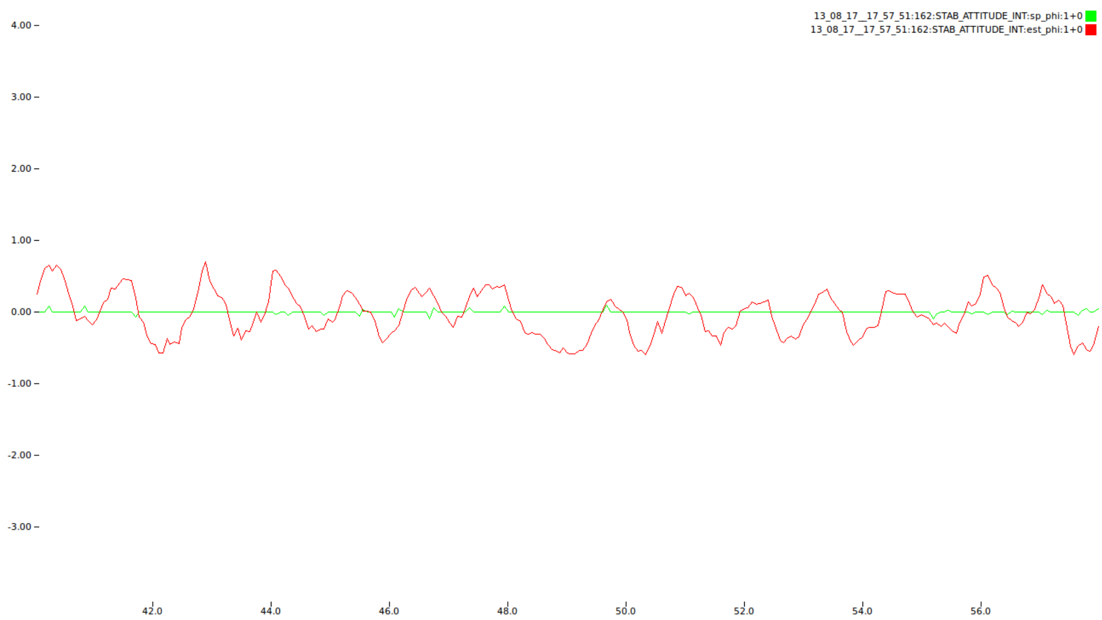


Figure 19: Roll angle estimation with PID_{windy1}

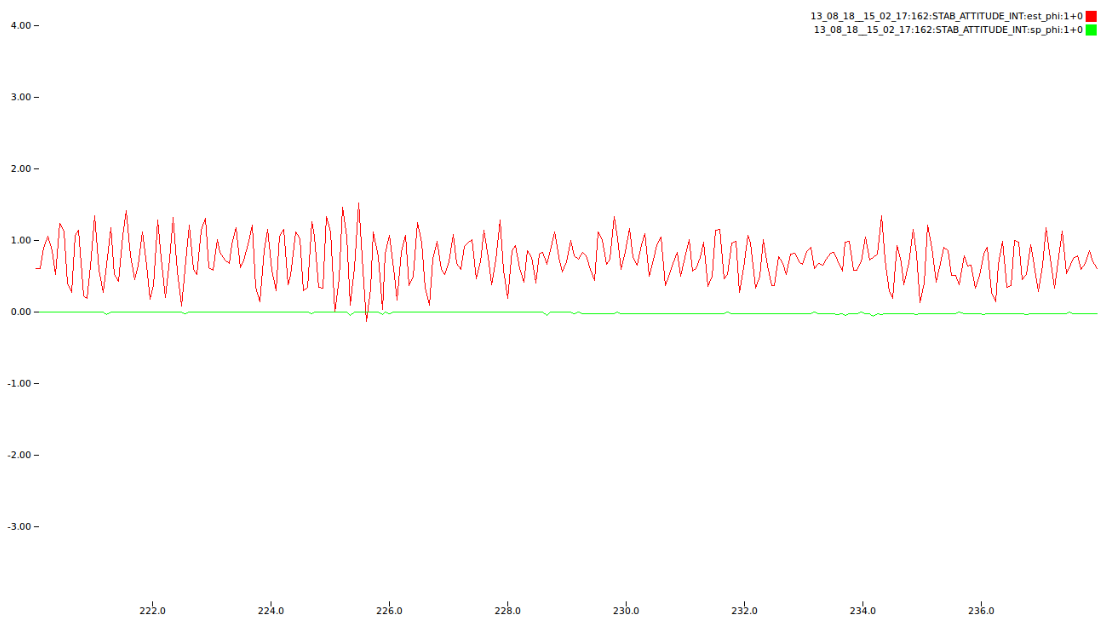


Figure 20: Roll angle estimation with PID_{windy2}

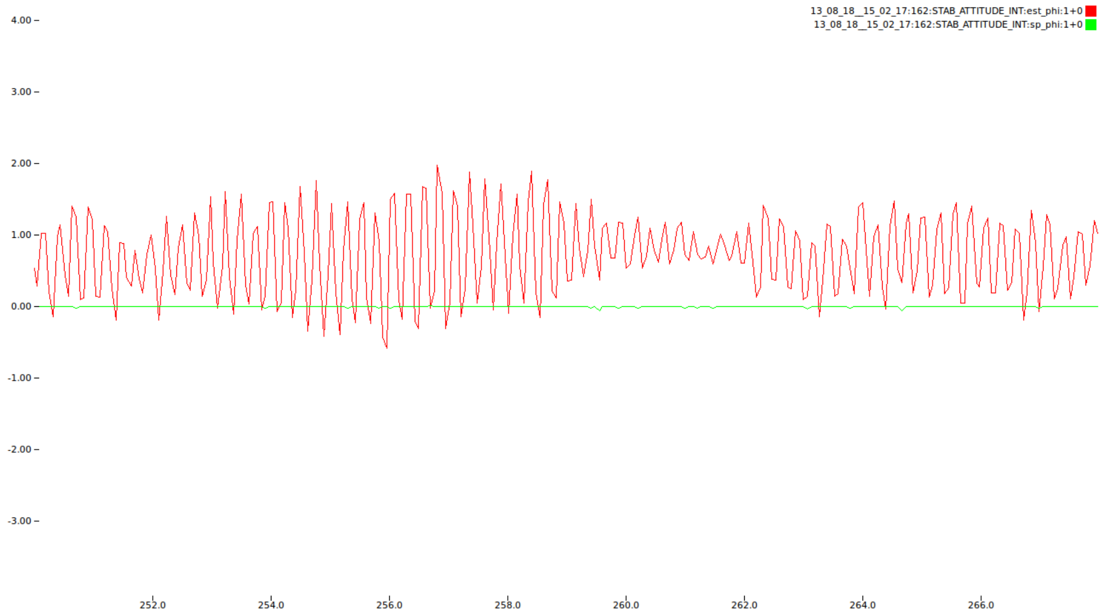


Figure 21: Roll angle estimation with PID_{windy3}

4.5 Compare Result from PID_{Windy} and $PID_{Original}$ in normal condition.

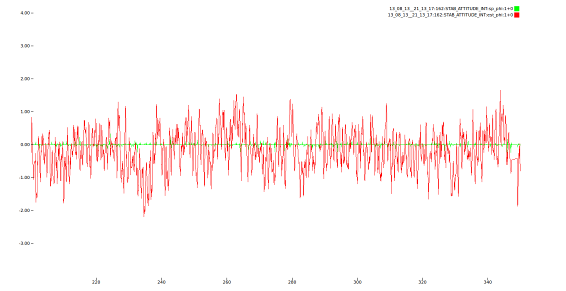


Figure 12: Roll angle estimation in normal condition with $D_{Original}$.	Figure 22: Roll angle estimation in normal condition with D_{Windy1} .

It seems that the PID_{Windy1} performed even better than $PID_{Original}$ in normal condition. What we are expecting is that PID_{Windy1} would bring along some drawbacks while it helped increase the stability in windy condition.

5. Conclusion

5.1 Outcome

Through the experiment, we have successfully found three PID_{Windy} ($\boldsymbol{PID_{Windy1}}$, $\boldsymbol{PID_{Windy2}}$, and $\boldsymbol{PID_{Windy3}}$) that can gives a much better performance than $PID_{Original}$ in both wind-simulation method 1 and method 2.

What is out of our expectation is that

(A) While $\boldsymbol{PID_{Windy2}}$ and $\boldsymbol{PID_{Windy3}}$ does decrease the stability in normal condition, $\boldsymbol{PID_{Windy1}}$ improve also the stability in normal condition.

That's to say, $\boldsymbol{PID_{Windy1}}$ is suitable for **both** windy condition and normal condition, by using this set of PID values, the quad-copter can perform quite well even under the influence of wind. The necessity for switching between PID_{Windy} and $PID_{Original}$ does not exist anymore.

(B) $\boldsymbol{PID_{Windy1}}$, $\boldsymbol{PID_{Windy2}}$ and $\boldsymbol{PID_{Windy3}}$ gives out almost the same performance in wind-simulation method 2 (the curves are bounded in 2 degrees).

If $\boldsymbol{PID_{Windy3}}$ or $\boldsymbol{PID_{Windy2}}$ performs better than $\boldsymbol{PID_{Windy1}}$, we can still treat $\boldsymbol{PID_{Windy1}}$ as the $\boldsymbol{PID_{Original}}$ that give overall good performance in both normal condition and windy condition, and declare that $\boldsymbol{PID_{Windy2}}$ gives better performance over $\boldsymbol{PID_{Windy1}}$ in windy condition but it's unstable in normal condition. But this did not happen; it seems that their performances are quite close in wind-simulation method 2.

5.2 Analysis about the outcome

There're multiple possible causes for this unexpected outcome.

1. **It's hard to find the 'optimal' PID values.**

This experiment is based on the **Assumption II** that 'we're already using the best PID parameters in normal condition'. If the set of PID parameters in normal condition we're using is not an optimized one, it's possible that when seeking for PID_{Windy} we will find a PID_{Windy} that provides overall better performance. If we have mastered some advanced PID optimization techniques, this problem can be avoided.

2. **Wind is too variable, the simulation method 1 does not necessary provide good PID values in simulation method 2.**

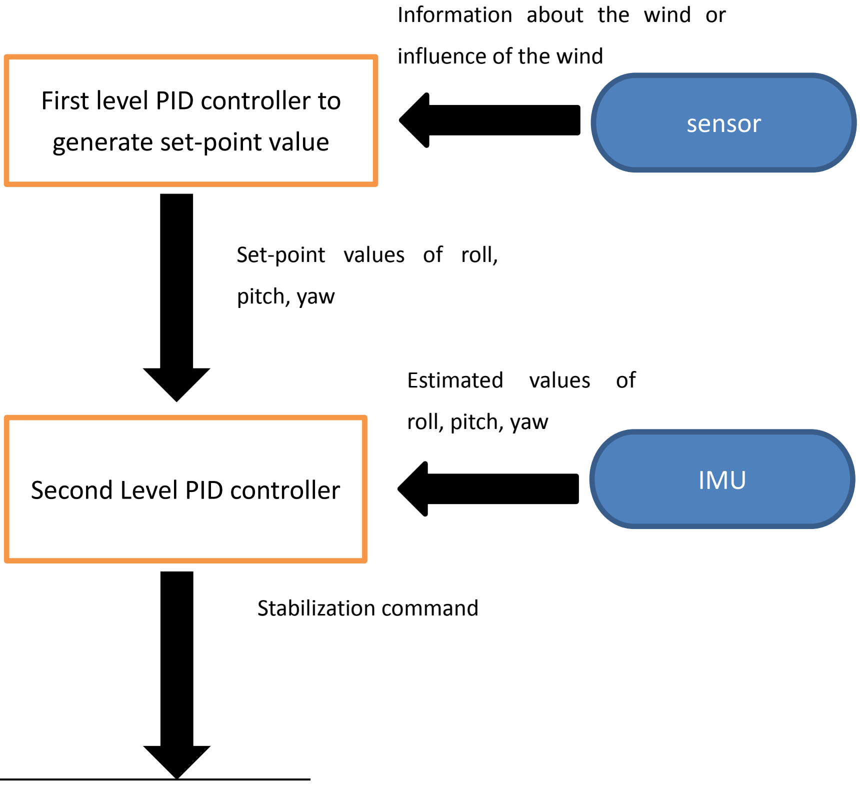
As it's known to all, wind is one of the most variable things of the world. This experiment is based on the thinking that wind will cause large deviation, so shorted transient time is needed for a better performance, and we did achieve shorter transient time by doing wind simulation method 2. What we did wrong is we didn't take the variability of wind into consideration, the deviation of the quad-copter's attitude comes very quickly and not like in the simulation method 1, so the set of PID values that do well does not necessarily get the best performance in simulation method 2.

Therefore, the procedure of finding PID_{Windy} should also be carried out in simulation method 2. And the Matlab simulation has to also consider the variability of deviation that can happen at any time.

5.3 Possible Solution

Experiencing the failure, we propose a possible solution to give quad-copter a better stability in windy condition.

We can construct a cascade PID controller on the quad-copter's attitude, rather than simply using only roll, pitch yaw angle. When wind is influencing the quad-copter, the first level PID controller can give out the set-point for roll, pitch and yaw angles, so the second level PID controller will finally compute the control signal, rather than in the old way, only stick the set-point of roll, pitch and yaw angle at 0.



---

[i] Hoffman, G.; Huang, H., Waslander, S.L., Tomlin, C.J. (20–23 August 2007). "Quadrotor Helicopter Flight Dynamics and Control: Theory and Experiment". *In the Conference of the American Institute of Aeronautics and Astronautics*. Hilton Head, South Carolina.

[ii] John S.Denker, "Coordinate Systems, Basis Sets, Reference Frames, Axes, etc.", available: http://www.av8n.com/physics/coords.htm#main-ned

[iii] Paparazzi Project: Paparazzi is a free and open-source hardware and software project intended to create an exceptionally powerful and versatile autopilot system for fixedwing aircrafts as well as multicopters by allowing and encouraging input from the community.
http://wiki.paparazziuav.org/wiki/Main_Page

[iv] Kiam Heong Ang, Gregory Chong, *Student Member, IEEE*, and Yun Li, *Member, IEEE, "PID Control System Analysis, Design, and Technology"*, IEEE TRANSACTIONS ON CONTROL SYSTEMS TECHNOLOGY, VOL. 13, NO. 4, JULY 2005, pp 559 – 576

[v] Paparazzi Project Control Loops,[accessed] 18th July 2013, [avaible]http://paparazzi.enac.fr/wiki/Control_Loops#Horizontal_Control

[vi] Jun Li, Yuntang Li, ”Dynamic Analysis and PID Control for a Quadrotor”, Proceedings of the 2011 IEEE International Conference on Mechatronics and Automation, August 7 - 10, Beijing, China.

[vii] S. Bouabdallah, R. Siegwart, “Design and Control of a Miniature Quadrotor”, in *Advances in Unmanned Aerial Vehicle, 2007* Springer, Printed in Netherlands, chapter6, pp171 - 210