Results and comparision between different control algorithms for a quadrotor using ArduPilot framework

- **Nguyen Anh Quang[1]**
- **Emmanuel Grolleau[2]**
- **Ngo Khanh Hieu[1]**

[1]Ho Chi Minh City University of Technology,VNU-HCMUT

[2]LIAS, ISAE – ENSMA, France

(Manuscript Received on July 13th, 2015; Manuscript Revised October 16th, 2015)

ABSTRACT:

Determining the most suitable control algorithm for a system is not an easy task. In theory, each controller has its own advantages and disadvantages comparing to the others. However, in the real world, the behavior of the controller also depends on many other factors such as the calculating ability of the control board, the accuracy of the sensors, the way the hardware communicate with the others, etc. In order to find the pros and cons of each control algorithm in the real world, each of them has to be tested and then comparing their results. This article presents a simple way to test the behavior of various control algorithms, with the quadrotor as the control target and ArduPilot is the framework to create the firmware carrying multi controllers. At the end of this article, the results of 3 control algorithms: Original PID of ArduPilot, new developed PID and Integral Backstepping will be presented and compared. These data is created by using Software In The Loop simulation (SITL), a tool provided by ArduPilot to test the new developed firmware.

Key words: *ArduPilot, control algorithm, quadrotor, PID, Integral Backstepping*

1. INTRODUCTION

Quadrotor is a six degree of freedoms system which is only controlled by four fixed-pitch equally-space rotors. In other words, even though the mechanically design is simple [1], this flying system is underactuated. The calculating for controlling this system will therefore be complicated. In theory, there are several control algorithms which is suitable for a quadrotor such as PID, Adaptive Control, Integral Backstepping [2], nonlinear H_∞ [3] or LQR controllers [4]. .However, there is no optimized controller for this system. Since each method has its pros and cons, the control algorithm for a quadrotor should base on the environment of the real system as well as its objectives. A controller, which can has the ability to change the control method in specific situations and desires will therefore be the best solution in this case. In order to experience the pros and cons of control methods, we decide to use the ArduPilot, a very popular framework used to create the firmware for the autonomous unmanned system, as the framework

to develop a module to integrate new control algorithms for the quadrotor. Using SITL simulation, we can verify that this module is good enough for taking the experiment with the real system and give us some ideas about the good and bad side of the integrated controllers.

2. QUADROTOR – FROM EQUATIONS TO INTEGRATED CODE

By default, there are several ways to create the integrated code to control a system. This article will present a solution suitable for complex systems, in this case a quadrotor. The basic of this solution is based on new tools which can transfer Simulink models into C code, as can be seen in figure 1.

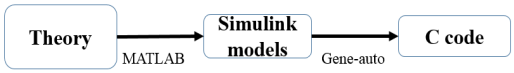

Figure 1. From theoay to C code

Using the Euler-Lagrange methods, the motion of the quadrotor plus frame is described by the following equations [6]:

$$\begin{cases} \ddot{\phi} = \dfrac{I_y - I_z}{I_x}\dot{\theta}\dot{\psi} + \dfrac{J_r}{I_x}\dot{\theta}\left(\Omega_1 - \Omega_2 + \Omega_3 - \Omega_4\right) + \dfrac{lb}{I_x}\left(\Omega_4^2 - \Omega_2^2\right) \\ \ddot{\theta} = \dfrac{I_x - I_z}{I_y}\dot{\theta}\dot{\psi} + \dfrac{J_r}{I_y}\dot{\phi}\left(-\Omega_1 + \Omega_2 - \Omega_3 + \Omega_4\right) + \dfrac{lb}{I_y}\left(\Omega_3^2 - \Omega_1^2\right) \\ \ddot{\psi} = \dfrac{I_x - I_y}{I_z}\dot{\phi}\dot{\theta} + \dfrac{d}{I_z}\left(-\Omega_1^2 + \Omega_2^2 - \Omega_3^2 + \Omega_4^2\right) \end{cases} \quad (1)$$

The controlled targets of the equation (1) are the Euler angles roll, pitch, yaw, which is represented by f, q and y ; meanwhile, the control outputs are the angular speed of the four motors. In order to test the equations above, they has been described by MATLAB Simulink model and then put in blocks with the principle shown in figure 2.

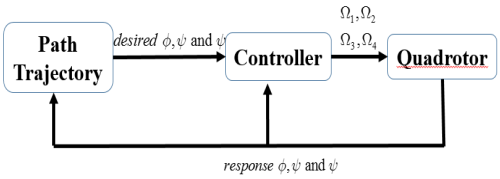

Figure 2. Blocks for the Simulink Model in MATLAB

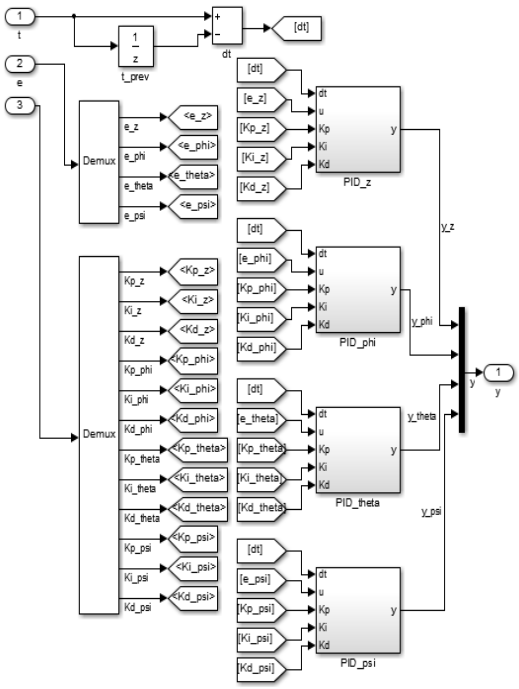

Figure 3. Simulink model for new PID controller

Based on the flight path or the inputs values from the users, the desired Euler angles will be created and then converted into the angular speed of each motor of the quadrotor. The Controller block can contain any kind of controller, as long as developers can describe it with Simulink model. This Controller block is then handled by the Gene-auto to create the necessary code. For example, figure 3 and figure 4 shows the Simulink model for the PID controller controlling the outputs of the quadrotor and the C code generated by Gene-auto.

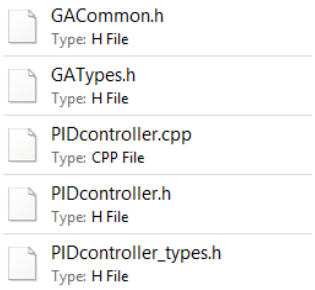

Figure 4. Code generated with GeneAuto

3. ARDUPILOT AND MODULE TO EMBED NEW CONTROLLERS

ArduPilot is one of the popular framework to create the firmware for an autonomous unmanned vehicle. One of the most important benefits of this framework comparing to others is that it has a multilayer structure, as described in figure figure 5. With this structure, this framework can support multiple control boards.

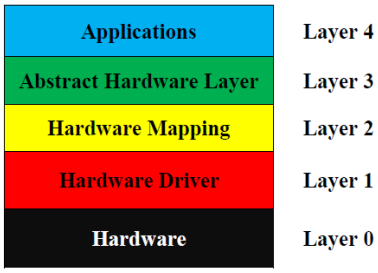

Figure 5. Multilayer structure of ArduPilot

In Vietnam, this framework is also very famous for developers, who have been familiar with boards such as APM2.5 or APM2.6 and the ground control station called Mission Planner. However, this article will focus more about the code and the modified to make this framework become multi-controllers, which is useful for users in the future.

The idea of this solution is simple, shown in figure 6. By default, ArduPilot has an original PID controller system, which control the rate of change of the Euler angles. In other words, this system handles the f, q and y by controlling $f^{\&}$, $q^{\&}$ and $y^{\&}$, PID control algorithm is used to make the real values of the system become as close as possible with the desired values. A new module has been created and embedded into the framework. The principle of the new add-in module is that users can change the using controller with just a single switch. By minimizing the modification, this module can use all of the advantages of the original code, for example the multilayer structure and the readiness for specific control boards, and still made the ArduPilot become a multi-controller framework.

As can be seen in this figure, if users choose to use the original controller, which is the default PID controller of ArduPilot mentioned above, nothing will change and the calculation process will be the same with the original code. However, when users decide to use a new controller, the calculating process will be changed and new control outputs will be generated based on the chosen control algorithm.

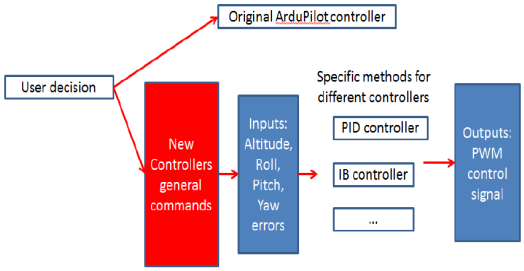

Figure 6. General idea of the new add-in module

4. ARDUPILOT AND CREATED MODULE TO EMBED NEW CONTROLLERS

This article will focus on introducing two of the control algorithms which have been successfully embedded into ArduPilot framework using the solution above.

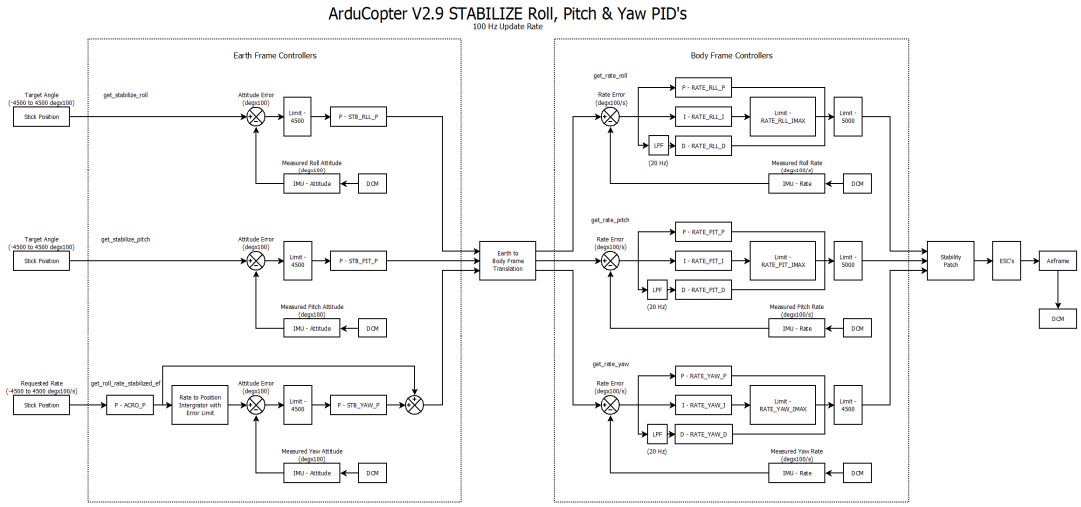

Figure 7. ArduPilot original PID controller

These results not only confirm the availability of the add-in module but also gives the comparison required to get the pros and cons of each controller with the quadrotor.

Unlike the original PID controls the rate of change of the Euler angles $\dot{f}$, $\dot{q}$ and $\dot{y}$, the new PID controller in figure 3 calculates the angular speed of motors based on the Euler angles f, q and y. The differences between the two control algorithms are small, however, by changing from the rate of change into the Euler angles, new PID controller reduces the amount of calculation required. This conclusion can be concluded according to the comparison between figure 7 and figure 3 above. In fact, as mentioned, both control algorithm has its benefits and drawbacks, and from the results shown in part 4, the original controller has better responses than the new PID controller.

"Backstepping control is a recursive algorithms that breaks down the controller into steps and progressively stabilizes each system" [2]. By adding an Integrator into the system to increase its robustness, the controller will become Integral Backstepping, which will not only work well with the dynamic of a quadrotor [5] but also make it is more stable with the disturbances [2]. Figure 8 introduces the IB controller used for a quadrotor.

With the definitions in equations (2), the motion equations of the quadrotor in case using the Integral Backstepping control algorithm will become equation (3). In equations (2), the values of c and λ are the control constants of the control algorithm; meanwhile, e is the error between the desired values and the real Euler angles respectively [6].

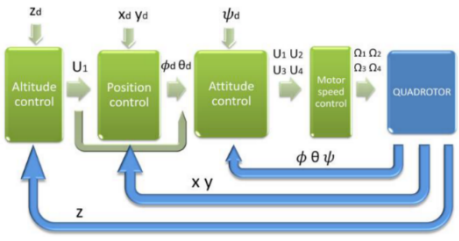

Figure 8. IB controller for a quadrotor

$$\begin{cases} u_x = \cos\phi\sin\theta\cos\psi + \sin\phi\sin\psi \\ u_y = \cos\phi\sin\theta\sin\psi - \sin\phi\cos\psi \\ \Omega_r = -\Omega_1 + \Omega_2 - \Omega_3 + \Omega_4 \\ U_1 = \dfrac{m}{\cos\theta\cos\phi}\left[g + \left(1 - c_7^2 + \lambda_4\right)e_7 + \left(c_7 + c_8\right)e_8 - c_7\lambda_4\chi_4 + \ddot{z}_d\right] \\ U_2 = I_x\left[\left(1 - c_1^2 + \lambda_1\right)e_1 + \left(c_1 + c_2\right)e_2 - c_1\lambda_1\chi_1 + \ddot{\phi}_d - \dot{\theta}\dot{\psi}\dfrac{I_y - I_z}{I_x} + \dot{\theta}\dfrac{J_r}{I_x}\Omega_r\right] \\ U_3 = I_y\left[\left(1 - c_3^2 + \lambda_2\right)e_3 + \left(c_3 + c_4\right)e_4 - c_3\lambda_2\chi_2 + \ddot{\theta}_d - \dot{\phi}\dot{\psi}\dfrac{I_z - I_x}{I_y} - \dot{\phi}\dfrac{J_r}{I_y}\Omega_r\right] \\ U_4 = I_z\left[\left(1 - c_5^2 + \lambda_3\right)e_5 + \left(c_5 + c_6\right)e_6 - c_5\lambda_3\chi_3 + \ddot{\psi}_d - \dot{\phi}\dot{\theta}\dfrac{I_x - I_y}{I_z}\right] \end{cases} \quad (2)$$

$$\begin{cases} \ddot{\phi} = \dot{\theta}\dot{\psi}\dfrac{I_y - I_z}{I_x} + \dot{\theta}\dfrac{J_r}{I_x}\Omega_r + \dfrac{U_2}{I_x} \\ \ddot{\theta} = \dot{\phi}\dot{\psi}\dfrac{I_z - I_x}{I_y} + \dot{\phi}\dfrac{J_r}{I_y}\Omega_r + \dfrac{U_3}{I_y} \\ \ddot{\psi} = \dot{\theta}\dot{\phi}\dfrac{I_x - I_y}{I_z} + \dfrac{U_4}{I_z} \\ \ddot{X} = \dfrac{U_1}{m}u_x \\ \ddot{Y} = \dfrac{U_1}{m}u_y \\ \ddot{Z} = \dfrac{U_1}{m}\left(\cos\phi\cos\theta\right) - g \end{cases} \quad (3)$$

By using MATLAB Simulink, the model of the Integral Backstepping can be described as in figure 9 and then embedded into the framework of ArduPilot. Users can choose to use this algorithm by using the new add-in module.

5. RESULTS WITH NEW PID CONTROLLER IN SOFTWARE IN THE LOOP SIMULATION (SITL)

Software In The Loop is a tool provided by ArduPilot to developers, which can be used to test new firmware and new modifications, in this case a new module to embed new controllers. Unlike Hardware In The Loop (HITL), which uses the virtual inputs with the real board to experience the the response of the real Hardware in some specific cases, SITL uses both virtual environment and hardware. Table 1 gives a simple comparison between two types of simulation.

Table 1. HITL and SITL comparisons

	HITL	SITL
Pros	- Testing the real behavior of a real system in situations which cannot be done with real tests because of endanger. - The result is more reliable since it is the reaction of a real hardware. - More safety as well as saving more time and money comparing to real system test. - Can be done in early stage and with separated components as well as a whole system.	- Does not need a real hardware. -Saving more time and money than HITL. - Can test many different situations, some of which cannot be done in HITL. - Easier to implement.
Cons	- In some situations, creating the virtual inputs is not simple, need to understand clearly about the testing environment. - For some systems, the real hardware is not only complicated to acquire but also require a lot of space.	- The result is not as reliable as HITL since it is just the behavior of a virtual hardware. - Needs time to create as well as to validate the virtual hardware, comparing its behavior with the real one.

Using SITL with the same flight path, figure 10 and figure 11 introduces the results with the original PID controller of ArduPilot and the new PID controller.

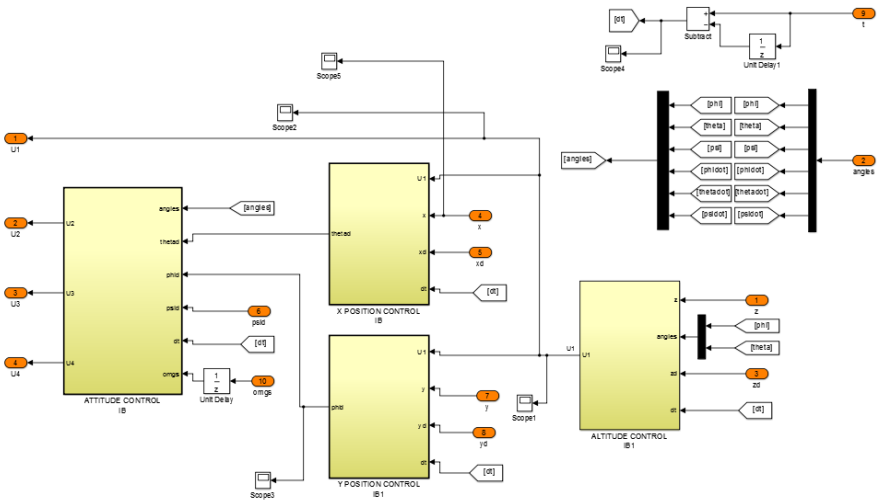

Figure 9. Integral Backstepping MATLAB Simulink model

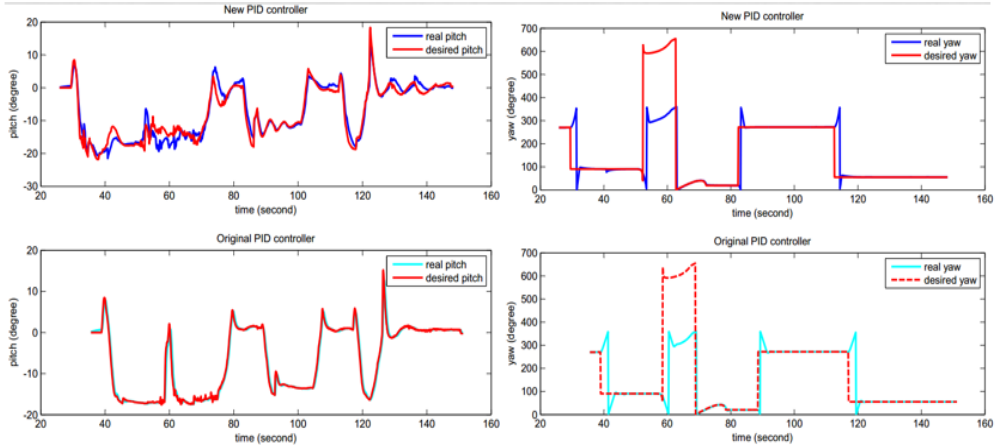

Figure 10. Pitch (left) and yaw (right) disired and response results with original and new PID

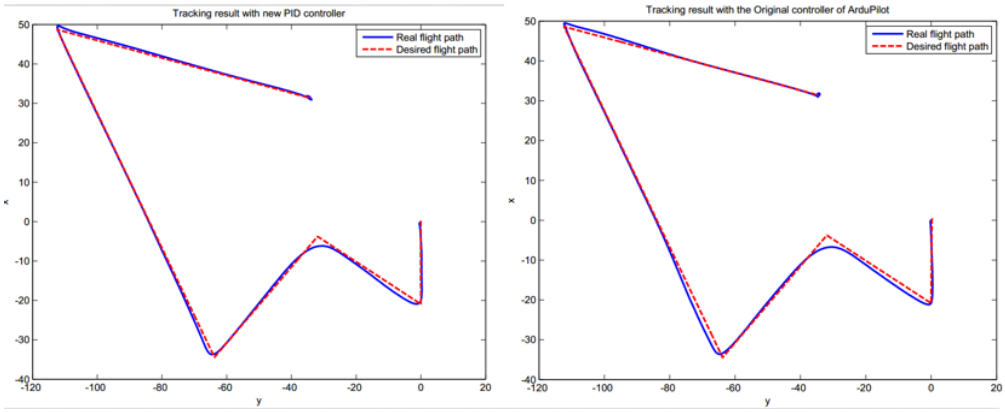

Figure 11. Tracking result with old PID (left) and new PID (right)

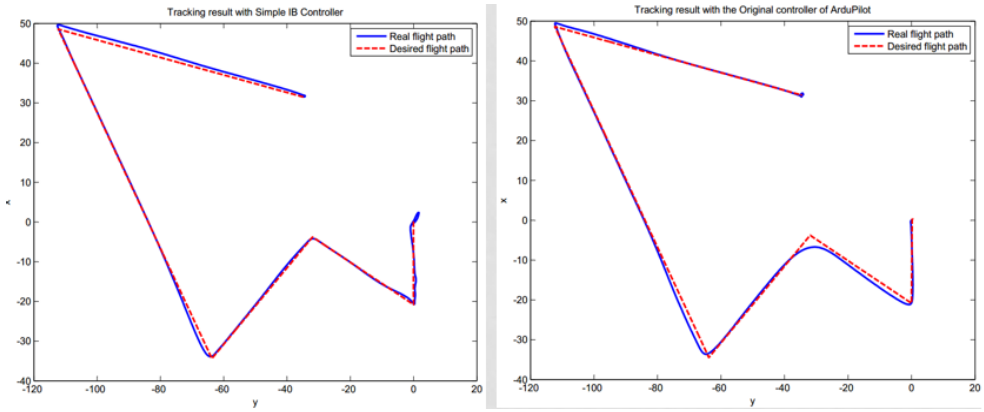

Figure 12. Tracking result with old PID (right) and IB controller (left)

The tracking ability of the new PID controller is as good as the old one (figure 10). Although there are still some errors, the new PID control algorithm still can drive the quadrotor back to the desired flight path. Figure 11 gives a more detail result. When comparing between the desired Euler angles and the response ones, it can be seen that the new PID controller results follow really close with the desired values. It is not as good as the old one, however it can be concluded that the new PID is steady enough for a real test.

6. RESULTS WITH INTEGRAL BACKSTEPPING IN SOFTWARE IN THE LOOP SIMULATION (SITL)

Using the same flight path with the Integral Backstepping, figure 12 and figure 13 demonstrate the results. Although the IB controller can trace the flight path well, there are some fluctuations as can be seen in figure 13. Nevertheless, as mentioned in the theory, IB controller has high robustness, which make the response of the system follow closely the desired values. Figure 14 give a more detail look for this conclusion.

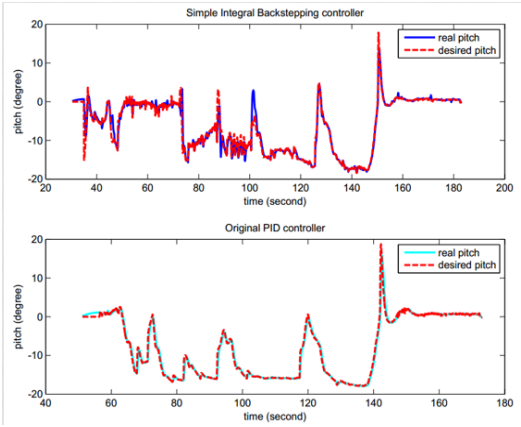

Figure 13. Pitch deired and response resutls with original PID and IB

7. CONCLUSIONS

With the results above, it is clearly that using the existence, open-source framework is one of the best solution to testing new control theory new modifications. With suitable changes, for example creating new add-in modules for necessary requirements, the modified firmware can use both the ready-to use structure of the original firmware and the benefits of the new code.

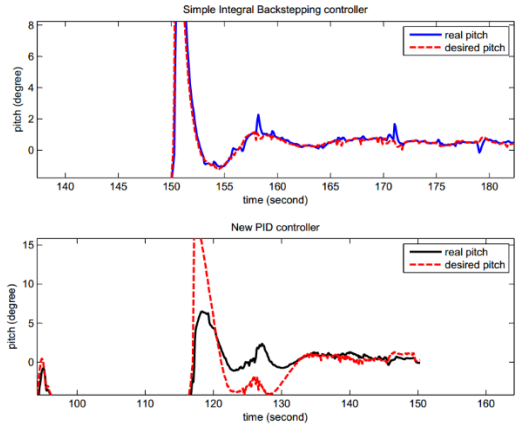

Figure 14. IB controller and new PID controller

PID control algorithm is one of the simplest one to control a system. It is well implemented to control various kind of system, one of them is the quadrotor. However, it is obviously that this is not the best solution and there are many other controller which is promising and need to be tested with the real things, not only by using the Simulink models. IB controller is one of them, which not only increases the robustness of the quadrotor but also has a very good tracking ability.

The result with the SITL simulation proves that a modified firmware built by ArduPilot is ready to test in real flight, which will give more results, especially the real response of the control board in real environment. By understanding the pros and cons of each controller in specific situation, a changeable controller, which is the optimized controller, can be implemented for a real quadrotor in the future.

So sánh và đánh giá khả năng điều khiển máy bay bốn chong chóng với các thuật toán khác nhau trên nền tảng ArduPilot

- **Nguyễn Quang Anh**[1]
- **Emmanuel Grolleau**[2]
- **Ngô Khánh Hiếu**[1]

[1]Ho Chi Minh City University of Technology, VNU-HCMUT

[2]LIAS, ISAE – ENSMA, France

TÓM TẮT:

Trên lí thuyết, mỗi thuật toán điều khiển đều có những ưu và nhược điểm đặc trưng. Trên thực tế, khả năng điều khiển cơ hệ còn phụ thuộc vào nhiều yếu tố khác của cơ hệ và hệ thống điều khiển. Trong trường hợp này, cách duy nhất để xác định chính xác

phản ứng của một hệ điều khiển là thử nghiệm trên hệ thống thực và đánh giá kết quả. Dựa trên việc sử dụng một hệ thống phức tạp là máy bay bốn chong chóng, bài báo này trình bày phương pháp đưa các hệ điều khiển khác nhau vào ArduPilot. Mô phỏng Software In The Loop đã được sử dụng để thực nghiệm 3 thuật điều khiển khác nhau: PID gốc của ArduPilot, PID tự phát triển và Integral Backstepping. Qua đó, ngoài việc xác định khả năng của hệ điều khiển, bài báo cũng nêu lên một vài kết quả bước đầu với các hệ điều khiển này, xác nhận lại lí thuyết đã biết của các thuật toán này, đồng thời là bước quan trọng để xác lập các hệ số điều khiển trước khi tiến hành bay thực.

Keyword: *ArduPilot, thuật điều khiển quadrotor, PID, Integral Backstepping*

REFERENCES

[1]. M. J. Cutler, *Design and Control of an Autonomous Variable-Pitch Quadrotor Helicopter*, M.Sc Thesis, Massachusetts Institute of Technology, 2012.

[2]. S. J. Andrew Zulu, *"A Review of Control Algorithms for Autonomous Quadrotors"* Open Journal of Applied Sciences, pp. 547-556, 2014.

[3]. G. Raffo, M.G.Ortega and F.R.Rubio, *"An integral predictive/nonlinear H∞ control structure for a quadrotor helicopter"* Automatica, vol. 46, pp. 29-39, 2010.

[4]. S. Bouabdallah, A. Noth et R. Siegwart, *«PID vs LQ control techniques applied to an indoor micro quadrotor»* IEEE/RSJ Internation Conference on Intelligent Robots and Systems, vol. 3, pp. 2451-2456, 2004.

[5]. Bouadallah.S et al, *«Full control of a quadrotor»* Intelligent Robots and Systems 2007. IROS 2007, pp. 153-158, 2007.

[6]. A. Benito, *«Flight Control and Navigation of a Quadcopter»* Poitiers, 2014.