Designing Flight Controller of Quadcopter Using Stm32 Microcontroller

Shubham Kadam[1], Sachin Ruikar[2],
[1]*M.Tech student, Walchand College of Engineering, Sangli*
[2] *Professor, Department of Electronics Engineering, Walchand College of Engineering, Sangli*

***Abstract:*. In order to achieve stable and responsive flight dynamics, designing a quadcopter flight controller with an STM32 microcontroller requires integrating sophisticated control algorithms, sensor fusion techniques, and effective hardware interfacing. This article offers a thorough method for creating and implementing a flight control system that makes use of the STM32 microcontroller, which is renowned for its potent computing capability and fast real-time performance. The choice and integration of sensors for accurate orientation and position monitoring, such as magnetometers, gyroscopes, and accelerometers, are important design elements. Proportional-Integral-Derivative (PID) controllers are among the flight control algorithms that have been fine-tuned for the STM32 architecture to guarantee a prompt and precise reaction to changing flying circumstances. Furthermore covered are the interfaces and communication protocols for telemetry, motor control, and user input. The resultant flight controller proves the STM32 microcontroller's efficacy in UAV applications with its sturdy performance throughout a range of test situations. The development of effective and dependable flight control systems for quadcopters and other unmanned aerial vehicles (UAVs) can benefit greatly from this study.**

Keywords: telemetry, flight controller, UAV, control algorithm

1. INTRODUCTION

The Drones may be roughly divided into two categories: multirotor and fixed-wing. Quadcopter or quadrotor is a traditional configuration that is commonly employed in multirotors. Four propellers are used by the quadcopter to generate lift and regulate the drone's dynamics.

There are four propellers; two spin in a clockwise direction and the other two in a counterclockwise direction. The drone may be maneuvered and directed in the desired direction by adjusting the rotors' speed.

The flight controller, airframe, motors, propellers, battery, ESC, transmitter, and receiver are the essential parts that make up the quadrotor. A flight controller collects data from sensors on board, such as a MEMS barometer and gyroscope, and uses control techniques to modify the data to achieve optimal flying. The flight controller modifies the motors' revolution rate in reaction to the information it receives from the receiver.

The flight controller may be divided into three categories: ground control station, software, and hardware components. The microprocessor, sensors, and other electronic parts make up the flight controller hardware. The drone is controlled and maneuvered by the flight controller software, which gathers data from sensors and applies certain control algorithms.

When the drone is in flight, it sends data to the ground control station, which shows flight data information such GPS coordinates, battery voltage, speed, etc. Drones may be controlled by a variety of flight controller systems, the most well-known of which being Pixhawk, Naza, Ardupilot Mega, and others. These platforms are hardware-specific or completely proprietary; therefore we are unable to alter them to meet our needs. To make modifications to the flight controller, one must understand how it operates. In order to achieve great customization, a flight controller is constructed utilizing an STM32 Microcontroller as the basis along with a number of additional sensors.

2. LITERATURE REVIEW

A survey on the autopilot system for tiny or micro unmanned aerial vehicles (UAVs) was provided by HaiYang Chao, YongCan Cao, YangQuan Chen, et al. in [1]. It talks about a variety of open-source, commercial, and research autopilot systems. Both the software and hardware perspectives are covered in their explanation. We analyze the sensor and microprocessor capabilities of many autopilots on the market. It states in the conclusion that the software design of a flight controller is significantly more important than the hardware.

The components of the flight controller and the UAV system are discussed by Emad Ebeid, Jie Jin, Martin Skriver, and others in [2]. The flight controller's software and hardware are also covered in this survey document. It contrasts the open source flight controller systems that are available based on their characteristics and, finds that Pixhawk is the greatest piece of gear because of its versatility and power. In a similar vein, it is determined that Arduino is a fully functional and superior piece of software.

A quadcopter flying controller based on the Raspberry Pi 3 was created by Sarah Pontes Madruga, Augusto de Holanda B. M. Tavares, Alisson Vasconcelos de Brito, Tiago Pereira Nascimento, and others in [3]. It describes the procedures and technical aspects of creating a flight controller. It describes the communication methods used by the motor and other sensors to connect to the flight controller. The article ends by stating that creating custom drone control algorithms and utilizing open-source hardware and software will be beneficial for a deeper comprehension.

An open source framework for drone flight controllers was presented by S. Sabikan, S. W. Nawawi, and colleagues in [4], allowing UAVs to be utilized for any outdoor application. The performance, stability, interference, and vibration findings of this paper's outside autopilot experimentation are presented. The conclusion is that the magnetometer may point in the incorrect direction due to magnetic field interference caused by motors, batteries, and ESCs. Elevation and horizontal location estimates based on accelerometers may deviate significantly from the true values when the flight controller experiences strong vibrations. In [5], Zhaolin Yang, Feng Lin, Ben M. Chen, and others examine the several autopilot systems that are out there and provide different control strategies to enhance the flight controller's performance and resilience. It also emphasizes the necessity of conducting Environmental Stress Screening (ESS) in order to ensure the flight controller's long-term dependability. For example, temperature cycling tests and random vibration tests

3. METHODOLOGY

An organized and thorough process is used in the design of the flight controller for a quadcopter that uses an STM32 microcontroller to guarantee excellent performance, dependability, and efficiency.

The STM32 microcontroller was chosen at the start of the process because of its powerful processing power, wide range of peripheral support, and real-time performance advantages. Its architecture makes it perfect for flight control applications by enabling the quick execution of control algorithms and effective processing of sensor data.

Subsequently, a selection of sensors was made with care, comprising magnetometers, gyroscopes, and accelerometers, to offer position monitoring and thorough orientation. To enable precise tracking of the quadcopter's motions, several sensors were incorporated into the system. The data from this many sensors was combined using sensor fusion techniques, which improved the precision and dependability of state estimates.

The creation of control algorithms, especially the Proportional-Integral-Derivative (PID) controllers, forms the basis of the flight control system. These algorithms were created to modify motor speeds in response to real-time sensor data, thereby controlling the quadcopter's responsiveness and stability. The design of the STM32 was carefully considered while optimising the PID controllers to ensure that they could operate fast and effectively, making short modifications to sustain steady flight.

In order to improve communication between the STM32 microcontroller and the sensors, motors, and other peripherals of the quadcopter, functional hardware interfaces were designed. This required creating and putting into practice communication protocols that guarantee timely and accurate data transmission. The motor control interfaces were especially important since they needed to provide accurate and consistent signalling in order to change the quadcopter's thrust and direction.

Using STM32 programming tools and libraries, the flight control software was created, taking use of the microcontroller's capabilities to regulate algorithm execution and handle sensor data processing. The most important operations, such receiving sensor data and updating motor control, were carried out promptly by using a real-time operating system (RTOS) to arrange and prioritise jobs.To assess the flight controller's performance, extensive testing was done under a variety of flight conditions. These tests were conducted in both controlled indoor and real-world outdoor contexts to verify the system's ability to handle various dynamic scenarios. The test results were carefully examined to find any problems with performance or potential areas for development.

Iterative enhancements were made to the hardware interfaces and control algorithms based on test findings. To improve system responsiveness, stability, and general performance, changes were made. The flight controller was enhanced with extra features including automatic tuning processes and fail-safes to make it more dependable and user-friendly.

The flight controller was created to match the strict requirements needed for reliable and successful quadcopter operation by adhering to this thorough process, proving the STM32 microcontroller's usefulness in UAV applications.

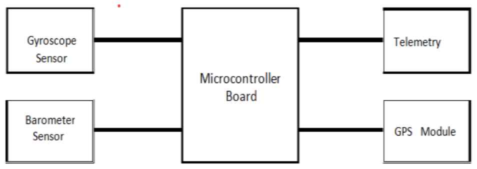

Fig.1 Block Diagram of flight controller

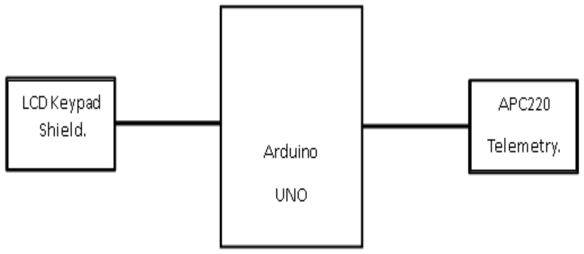

Fig.2 Block Diagram of Telemetry System

Programming the STM32 Blue Pill involves loading the STMduino library into the Arduino software for compatibility and choosing the right board using the Arduino IDE software.

First, by configuring the modules, the sensors—GPS, barometer, and gyroscope—are checked separately. The STM board may receive signals from the receiver in two different forms. PWM is used for one, while PPM for the other. A series of square wave pulses are sent from the transmitter to the receiver in order for both devices to function. The sole distinction is that while PPM utilizes a single wire for each of the six channels, PWM uses six wires for each of the six channels. In order to simplify wiring, PPM signals are utilized in communication. The gyroscope provides readings for roll, pitch, and yaw. The gyroscope's initial settings are saved onto the board's EEPROM memory when the quadcopter is armed.

The drone is stabilized using the stored values as a foundation.

A. PID Controller

When there are unexpected disruptions, the flight controller applies a control algorithm to steady the drone. The drone cannot steady itself if there is no control algorithm. For stability, the PID control method is employed. Proportional Integral Derivative algorithm is referred to as the PID. Term P provides a gain to ensure that there is no mistake by calculating the difference between the setpoint and the process variable, or error. In order to ignore error, Term I incorporates historical error values across time. To minimize inaccuracy, term D calculates the present rate of change and differentiates across time. Using the recorded gyro values as a foundation, the control algorithm attempts to minimize the error.

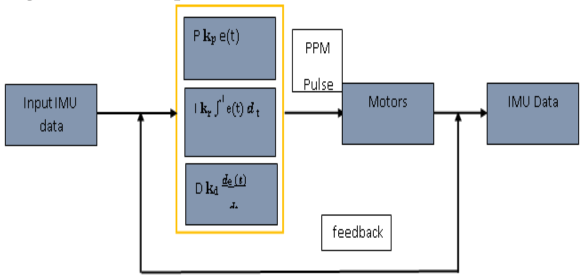
Input IMU data
P $\mathbf{k}_p$ e(t)
I $\mathbf{k}_r \int^{t}$ e(t) d_t
D $\mathbf{k}_d \frac{de(t)}{dt}$
PPM Pulse
Motors
IMU Data
feedback

Fig.3 Block Diagram of PID

The controller receives the original imu values (roll, pitch, and yaw) as input. Based on the current state and the command, it calculates the error rate and lowers the error by controlling the motors by providing ppm pulses to the esc's, which in turn controls the motors and adjusts the quadcopter's imu angles. In this manner, it ignores the wobbling problems and stabilizes the drone.

The drone will always attempt to stabilize by performing the PID calculations in the main program loop, but when the transmitter delivers signals to the receiver, an interrupt will begin to act, interrupting the PID computations. PID continues after interrupt begins to operate by detecting the rising edge pulse upon detection of the receiver signals. As soon as a falling edge pulse is detected, the main program loop will continue and an interrupt will be started based on the falling time.

I. Voltage Divider

Knowing the battery voltage of a quadcopter is crucial because if it runs out while in flight, the drone may crash and malfunctioning parts may result. Therefore, understanding the battery voltage is essential. The flight controller uses a voltage divider circuit to determine the battery voltage. The high voltage number is changed to a lower one via the voltage divider. Since STM32 is only capable of 3.3V, we must convert high voltage to lower voltage; else, the board would be damaged. So, a circuit with a voltage divider is utilized. The battery has a maximum output voltage of 12.6V. The max is converted using two resistors.

12.6 V to less than 1.1 V. The circuit schematic is displayed below

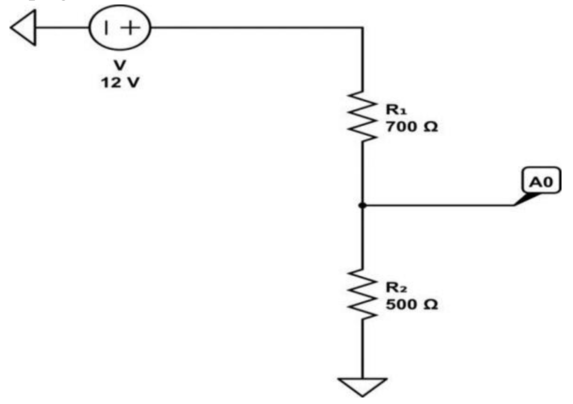

Fig.4 Voltage Divider

The STM32 board's analog pin is linked to the output voltage pin. The output voltage will be represented as a logarithmic value between 0 and 4095. The formula will be used to convert this analog value to the original voltage. The following is a theoretical formula to determine the output voltage:

Vout =Vin $R2$

$R1+R2$…………..(1)

The following formula will be used in the software to convert the analog value on the STM board to its original voltage.

Vin $Analog\ value$ (2)

112.81

In addition, the altitude and location coordinates are obtained through the interface of the barometer, GPS, and telemetry sensors. For monitoring, the data collected in flight are transmitted to the ground telemetry. The Arduino UNO is used to power the ground telemetry system. The UNO is interfaced with an LCD display to provide for visual flight monitoring.

4. RESULT

Despite using the control method, there were a lot of drifts throughout the test flight because it was unstable. The gain values were the source of the issue. After some fine-tuning, the drone gradually became stable. Once the drone is fully stabilized, the gain parameters are adjusted. Below are the final gain values following tweaking.

P Gain = 1.0 1 Gain= 0.02

D Gain= 10.0

Because of the electronic speed controller interference, the battery voltage readings were a little bit erroneous. A filter is employed to make up for the mistake. The filter's formula is shown below.

Battery Voltage = (Previous loop voltage 0.92) + (analog value/1410.1)

The MS5611 barometer has a high light sensitivity. It provides a about 5% inaccuracy rate when exposed to light. The barometer sensor is housed within the shell with holes punched on the sides to allow air to travel through in order to ignore this. The graph below displays the comparison under normal conditions as well as under conditions of light exposure. (For informational purposes only) The test was conducted at a 20-meter height

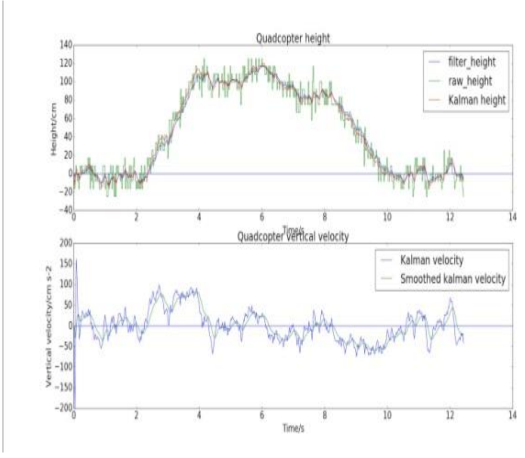

Fig.5 Data of Barometer

To verify correctiveness, the flight controller's gyroscope measurements are collected and put to the test. The gyroscope provides readings for roll, pitch, and yaw. The quadcopter was tilted in different directions and degrees to collect the data. The obtained data are then represented and viewed in the graph that is seen below

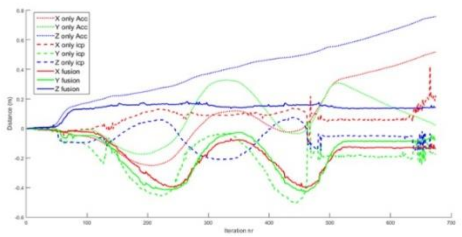

Fig.5 Data of IMU

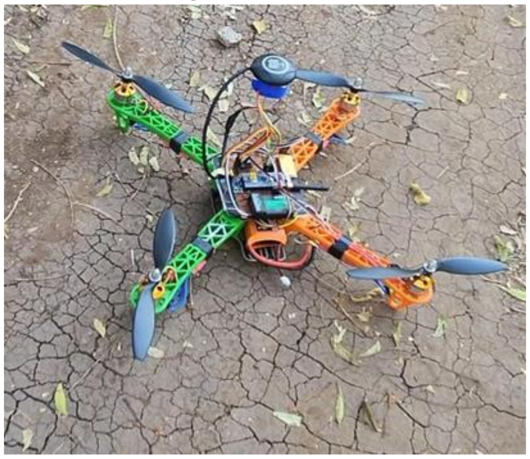

Fig.6 Flight Controller with Quadcopter

5. CONCLUSION

It has been determined that if there is a loss of telemetry contact with the unmanned aerial vehicle (UAV) during first-person view (FPV) flight, the pilot experiences a total loss of spatial awareness in the surrounding environment. Changing the flight

The controller may be set to either NAV RTH (wing) or GPS Rescue (copter) mode. In these modes, the device will consistently return to the launch zone. When a video connection is established, the pilot can easily switch back to manual control using the remote control.

The feasibility of achieving accurate autonomous return to the initial location only relying on a GPS receiver for rotor-type UAVs and fixed-wing aircraft, without the need for a magnetometer and barometer, is demonstrated. As a result, the design was simplified and its cost was reduced.

It is mentioned that an important requirement for the return is stable connection of GPS receiver with the number of satellites not fewer than those defined in the corresponding parameters. Furthermore, in the event of a communication breakdown, the flying wing has the capability to sustain a prolonged flight along a predetermined path until communication is reestablished. The quadrocopter is equipped with a failsafe mechanism that automatically shuts down the motors and executes a controlled descent in the event of a communication failure lasting 1-2 seconds.

To ensure reliable functioning of the GPS Rescue mode, it is necessary for the copter to utilise the stabilisation mode (Angle) with the accelerometer on, maintain minor tilt angles during flight, and operate in cloudy weather conditions. A study revealed that as the degree of inclination of the copter increases, the number of satellites caught by the GPS receiver decreases. Hence, it is not recommended to utilise GPS Rescue mode during Acro, 3D, Horizon flying modes, and when doing flips.

REFERENCES

[1] Hai Yang Chao, Yong Can Cao, and Yang Quan Chen, "Autopilots for Small Unmanned Aerial Vehicles A Survey, "International Journal of Control, Automation, and Systems, Vol. 8, No.1, pp.36-44, 2010.

[2] Emad Ebeid, Martin Skriver, Jie Jin, "A Survey on Open-Source Flight Control Platforms of Unmanned Aerial Vehicle," Euromicro Conference on Digital System Design, 2017.

[3] Sarah Pontes Madruga, Augusto de Holanda B. M. Tavares, Alisson Vasconcelos de Brito, Tiago Pereira Nascimento, "A Project of an Embedded Control System for Autonomous Quadrotor UAVs," Brazilian Symposium on Robotics, 2018.

[4] S. Sabikan, S. W. Nawawi, "Open-Source Project (OSPs) Platform for Outdoor Quadcopter. Journal of Advanced Research Design," Vol. 24, No. 1, pp. 13-27, 2016.

[5] Zhaolin Yang, Feng Lin, Ben M. Chen, "Survey of Autopilot for Multi-rotor Unmanned Aerial Vehicles," IECON 2016-42nd Annual Conference of the IEEE Industrial Electronics Society, pp. 6122-6127,2016.

[6] H. Lim, J. Park, D. Lee, and H. J. Kim, "Build your own quadrotor. Open-source projects on unmanned aerial vehicles," IEEE Robotics & Automation Magazine, vol. 19, no. 3, pp. 33-45, 2012.

[7] Sangook Cho, Sanghyuk Park and Keeyoung Choi, Choi, "Autopilot Design for a Target Drone using Rate Gyros and GPS." International Journal

of Acronautical and Space Science, Vol 13, No. 4, pp. 468-473, 2012.

[8] Atheer L.. Salihland and M. Moghavvemi, "Flight PID Controller Design for a UAV Quadrotor," Scientfic Research and Essays, Vol. 5, No. 23, pp. 36603667, 2010.

[9] C. Nam and S. Danielson, "Development of a Small UAV with real-time v video surveillance," American Society for Engineering Education, 2011.

[10] H. Wu, D. Sun, and Z. Zhou, "Micro air vehicle: Configuration, analysis, fabrication, and test," IEEE/ASME Trans. on Mechatronics, vol. 9, no. 1, pp. 108-117,2004

[11] R. Beard, D. Kingston, M. Quigley, D. Snyder, R. Christiansen, W. Johnson, T. Melain, and M. Goodrich, "Autonomous v chide.