Jurnal Borneo Informatika & Teknik Komputer **p-ISSN:** 2808-8190
Vol. 5, No. 2, 2025. Hlm 11—21 **e-ISSN:** 2808-6600

Development of an Autonomous Quadcopter *Drone* Equipped With a *Waypoint* Navigation System

Muh. Taufiqurrohman[1*], Rasional Sitepu[1], Sinung Widiyanto[2], Moch. In'am Zayyid Romadhon[2]
[1]Program Studi Profesi Insinyur, Universitas Katolik Widya Mandala Surabaya
[2]Program Studi Teknik Elektro, FTIK, Universitas Hang Tuah
e-mail: [1]taufiqurrohman@hangtuah.ac.id

Diterima	Direvisi	Disetujui
16-10-2025	29-10-2025	06-11-2025

Abstract: *The development of quadcopter drone technology has progressed rapidly and has been utilized in various fields, including mapping, agriculture, surveillance, and scientific research. In the academic environment, drones play a role not only as objects of study, but also as a medium for learning and developing automated system technology. To support this, a flexible and autonomous drone platform is needed. ArduPilot as an open-source autopilot software, together with the Pixhawk flight controller, is a widely used solution because it supports various features such as waypoint navigation, auto takeoff, and return to launch. The system enables open, modular, and integrative development of UAVs with various sensors. This research aims to design and implement an autonomous quadcopter drone prototype based on ArduPilot and Pixhawk as a means of supporting research and practicum in the campus environment, in order to encourage cross-disciplinary collaboration and readiness to face the industrial era 4.0 and society 5.0*
Keywords: Drone quadcopter, autonomous, ArduPilot, Pixhawk, UAV

Abstrak: Era Perkembangan teknologi *drone quadcopter* mengalami kemajuan pesat dan telah dimanfaatkan dalam berbagai bidang, termasuk pemetaan, pertanian, pengawasan, dan penelitian ilmiah. Di lingkungan akademik, *drone* berperan tidak hanya sebagai objek kajian, tetapi juga sebagai media pembelajaran dan pengembangan teknologi sistem otomatis. Untuk mendukung hal tersebut, dibutuhkan *platform drone* yang fleksibel dan dapat dikendalikan secara *autonomous*. ArduPilot sebagai perangkat lunak autopilot *open-source*, bersama dengan *flight controller Pixhawk*, menjadi solusi yang banyak digunakan karena mendukung berbagai fitur seperti navigasi *waypoint*, auto *takeoff*, dan *return to launch*. Sistem ini memungkinkan pengembangan UAV secara terbuka, modular, dan integratif dengan berbagai sensor. Penelitian ini bertujuan untuk merancang dan mengimplementasikan prototipe *drone quadcopter autonomous* berbasis ArduPilot dan *Pixhawk* sebagai sarana pendukung riset dan praktikum di lingkungan kampus, guna mendorong kolaborasi lintas disiplin dan kesiapan menghadapi era industri 4.0 serta *society* 5.0.
Kata kunci: Drone quadcopter, autonomous, ArduPilot, Pixhawk, UAV

I. PENDAHULUAN

Perkembangan teknologi *drone quadcopter* saat ini semakin pesat dan banyak dimanfaatkan dalam berbagai bidang seperti pemetaan wilayah, pertanian, pengawasan, serta penelitian ilmiah. Di lingkungan akademik, *drone* tidak hanya menjadi objek kajian, tetapi juga alat bantu yang potensial dalam proses pembelajaran, praktikum, hingga pengembangan teknologi berbasis kontrol otomatis dan sistem tertanam (*embedded system*)(Budiharto et al., 2021). Oleh karena itu, kebutuhan akan *platform drone* yang dapat dikembangkan secara fleksibel dan dikendalikan secara *autonomous* (Suparta, 2024) menjadi penting bagi kampus - kampus yang ingin membangun kapasitas riset dan inovasi teknologi UAV. Salah satu sistem yang banyak digunakan dalam pengembangan *drone autonomous* adalah ArduPilot(Yusuf, 2024), sebuah perangkat lunak *open-source* autopilot yang mendukung berbagai jenis UAV termasuk *quadcopter*, *fixed-wing*, dan kendaraan darat maupun laut(Saroinsong et al., 2018). ArduPilot dapat bekerja optimal bersama *flight controller Pixhawk*,

yaitu perangkat pengendali penerbangan yang mendukung pemrograman misi otomatis seperti *waypoint navigation, auto takeoff*, dan *return to launch*(Setiawan et al., 2017). Sistem ini memungkinkan *drone* untuk beroperasi tanpa intervensi langsung dari operator selama misi berlangsung, sehingga cocok untuk diterapkan dalam skenario eksperimen dan misi riset kampus (Rivai et al., 2020). Kombinasi ArduPilot dan *Pixhawk* sangat cocok digunakan sebagai *platform* pengembangan UAV di lingkungan kampus karena selain bersifat terbuka dan dapat dimodifikasi, sistem ini juga mendukung integrasi dengan berbagai sensor dan perangkat eksternal(Aliane, 2024). Hal ini membuka peluang bagi mahasiswa dan peneliti untuk melakukan eksplorasi teknologi lebih lanjut, seperti navigasi berbasis GPS, pemrosesan data udara, dan penerapan algoritma kecerdasan buatan(Harjanto, 2020).

Di sisi lain, pengembangan *drone* di kampus juga memiliki peran penting sebagai sarana kolaborasi antar disiplin ilmu, seperti teknik elektro, teknik komputer, informatika, dan geospasial(Hartono, 2018). Dengan tersedianya *prototipe drone quadcopter* yang dapat dikendalikan secara *autonomous*, institusi pendidikan dapat menyediakan infrastruktur praktikum dan riset yang lebih maju dan relevan dengan kebutuhan industri 4.0 dan society 5.0. (Qian et al., 2018). Berdasarkan latar belakang tersebut, penelitian ini bertujuan untuk merancang dan mengimplementasikan sebuah *drone quadcopter* yang dapat dikendalikan secara *autonomous* menggunakan ArduPilot dan *flight controller Pixhawk*, guna menunjang pengembangan dan eksperimen teknologi UAV di lingkungan kampus.

II. METODE PENELITIAN

Metodologi penelitian ini digunakan untuk merancang dan menguji sebuah *drone quadcopter autonomous* yang dikendalikan menggunakan *software* ArduPilot dan *flight controller Pixhawk*. Penelitian ini dilaksanakan dengan pendekatan desain eksperimen yang mencakup beberapa tahapan, mulai dari perancangan sistem, implementasi, hingga pengujian dan analisis hasil.

1. Desain Sistem

Proses desain sistem ini dimulai dengan pemilihan *hardware* dan *software* yang akan digunakan dalam pembuatan *drone quadcopter*. *Hardware* yang digunakan mencakup:

a. ***Quadcopter Frame*:** Memilih *frame quadcopter* yang sesuai dengan ukuran dan kapasitas beban yang dibutuhkan untuk mengangkat komponen elektronik. Rencananya untuk *drone quadcopter* ini nanti akan memakai *frame* dengan tipe F450 dengan tambahan kaki untuk melakukan pendaratan lebih aman. *Frame* ini memiliki bentuk lengan menjadi huruf X pada ujung papan tempat peletakan komponen. *Frame* ini berbahan plastik yang keras dan berukuran cukup tidak terlalu lebar(Budiharto et al., 2021). Terdapat 2 tingkat papan sekaligus pcb untuk *supply* tegangan listrik dari baterai ke esc dan motor.

b. ***Flight Controller*:** Menggunakan *Pixhawk* 2.4.8 sebagai *flight controller* utama. *Pixhawk* ini akan mengelola data sensor dan menjalankan sistem autopilot dari ArduPilot. *Pixhawk* yang nantinya akan menentukan titik kemiringan dari *drone,* kemudian menentukan pergerakan *drone,* dan juga menentukan serta mengolah komunikasi *drone* dengan *user* baik secara *autonomous* ataupun dengan remot(Yasa, 2022)(Sutikno & Darat, 2025). Data yang diterima *Pixhawk* juga akan ditampilkan pada *software* Mission Planner.

c. **Sensor:** Menggunakan GPS module untuk navigasi, IMU (*Inertial Measurement Unit*) untuk stabilitas, serta barometer untuk pengukuran ketinggian. Komunikasi *drone* dengan sensor akan terus dilakukan untuk mendapatkan nilai yang akurat selama *drone* menjalankan misi. *Drone* juga akan menampilkan hasil sensor pada tampilan Mission Planner.

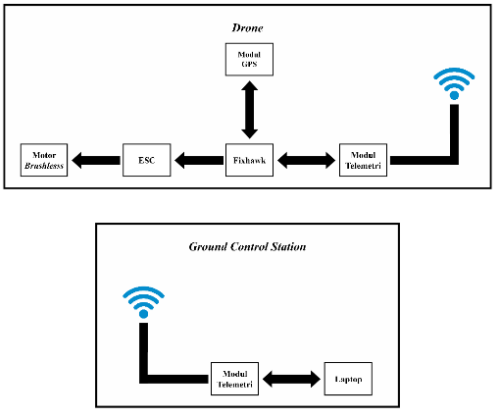
d. **Motor dan ESC (*Electronic Speed Controller*):** Motor *brushless* dengan ESC akan digunakan untuk penggerak utama *drone*. Kecepatan dan pergerakan Motor dan ESC ini ditentukan oleh *Pixhawk*. Kecepatan putaran motor meningkat atau menurun akan membuat kemiringan *drone* berubah sehingga dapat melakukan pergerakan maju, mundur, kesamping, *take-off,* ataupun *landing.*

Gambar 1. Layout komponen pada drone

Pemasangan GPS dilakukan pada sedikit lengan kanan depan. Hal tersebut dilakukan karena pemasangan menggunakan tiang yang menjulang keatas mengganggu aerodinamis dari *drone*. Pemasangan juga tidak dilakukan diatas *Pixhawk* dikarenakan akan menutupi port yang akan mengganggu proses bongkar pasang *socket* jika diperlukan. Pada sisi perangkat lunak, ArduPilot akan digunakan untuk mengendalikan *drone* secara otomatis(Haryanto et al., 2021). ArduPilot akan dihubungkan dengan Mission Planner sebagai *Ground Control Station* (GCS) untuk pemrograman misi dan pengaturan kontrol penerbangan(Efison, 2023).

2. Diagram Blok

Pada diagram blok dapat dilihat komunikasi dilakukan secara dua arah oleh *Pixhawk* pada *drone* dan laptop di *ground station*. Hal ini yang membuat *drone* dapat bekerja secara *autonomous*. Diawali dengan laptop adalah perantara *user* kepada *drone* untuk mengirimkan misi. Dengan Mission Planner *user* akan menentukan titik *waypoint,* menentukan ketinggian, serta memerintahkan untuk *take-off* dan *landing*(Tamtomi et al., 2016). Laptop dengan *Pixhawk* akan berkomunikasi melalui modul telemetri yang terpasang dikeduanya. Misi akan di*upload* melalui modul telemetri yang sudah terhubung sebelumnya. Pada *drone Pixhawk* akan meminta modul GPS untuk mengidentifikasi posisi letak dari *drone.* Kemudian GPS akan mengirimkan data nilai posisi *drone. Pixhawk* akan mencatatnya dan juga mengirimkan data tersebut kepada laptop melalui modul telemetri sehingga *user* dapat melihat keberadaan *drone.* Setelah *Pixhawk* mendapat perintah misi yang telah dikirimkan *drone* hanya perlu menunggu perintah mulai untuk melakukan misi.



Gambar 2. Diagram blok komunikasi *drone* dengan GCS

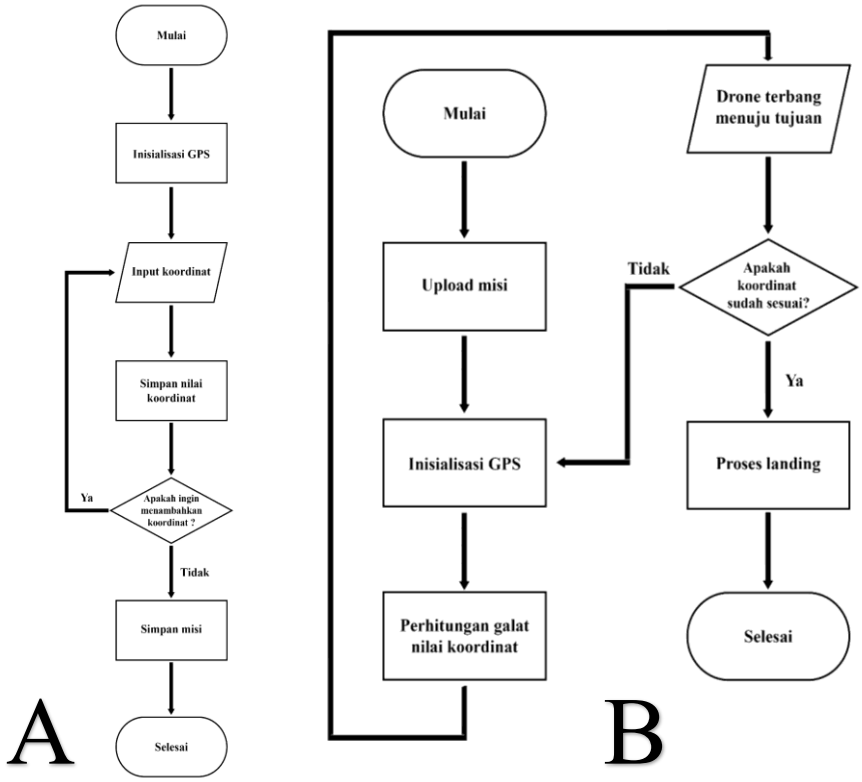
Pixhawk akan mengirimkan nilai PWM kepada ESC sehingga motor berputar sesuai dengan nilai PWM yang diberikan. Kecepatan putaran motor dikeempat sisi akan menentukan kendali *drone* disaat dia memulai *take-off*, mengubah posisinya (berputar), berjalan baik, mundur, ataupun kesamping, serta melakukan *landing. Pixhawk* yang akan mengolah data disetiap sensor untuk memastikan *drone* bergerak menuju titik *waypoint* pada misi. Ia akan bergerak dari perhitungan

galat yang terjadi pada pengolahan datanya.

3. *Flowchart*

Flowchart pertama (Gambar 3-a) menampilkan alur di *ground control station* dimana kita sebagai *user* yang memberikan perintah dengan menentukan titik – titik tujuan yang diinginkan sebelum dipelajari dan dikerjakan oleh *drone*. Proses ini dilakukan pada *software* Mission Planner Ardupilot dilaptop. Alur diawali dengan inisialisasi posisi oleh GPS setelah *drone* dan laptop terkoneksi. Laptop dan *drone* masing – masing menampilkan posisinya pada Mission Planner. Lalu *user* akan melakukan *input* titik koordinat dengan langsung menitik pada tampilan peta. Kemudian *user* memberikan perintah gerakan pada masing – masing titik. Nilai koordinat pada peta dapat dilihat di kolom *latitude* dan *longitude*. Selain itu, ketinggian dapat ditentukan pada kolom ketinggian. Jarak antar titik juga dapat dilihat pada kolom *distance*.

Setelah menentukan titik – titik yang dituju kita akan menyimpan data nilai koordinat. Lalu alur akan meyakinkan apakah akan menambahkan kembali titik *waypoint* atau lanjut. Jika dirasa ada yang perlu dirubah maka akan ditentukan ulang, namun jika dirasa cukup rute dapat disimpan ke penyimpanan laptop menjadi *file* rute yang dapat dibuka kembali dan juga di*upload* ke *Pixhawk* untuk dijalankan misinya oleh *drone* dan alur pun selesai dilakukan.



Gambar 3. (a) *Flowchart* alur perencanaan misi oleh user pada GCS (b) *Flowchart* alur *drone* mengolah misi yang diberikan

Pada *flowchat* kedua (Gambar 3-b) menampilkan proses yang dilakukan oleh *drone* setelah proses *upload* dan perintah titik tujuan diterima. Diawali dengan *upload* misi yang telah diterima dan diproses oleh *Pixhawk* kemudian *drone* akan menginisialisasi posisinya. Dari titik yang diminta dan posisinya sekarang, *drone* akan melakukan perhitungan dan didapatkan galat yang harus di eksekusi. Galat tersebut akan membuat *Pixhawk* mengirimkan pwm ke ESC yang membuat motor berputar. Perputaran motor menjadikan *drone* bergerak terbang menuju titik tujuan. Alur akan terus berulang dengan inisialisasi koordinatnya dari GPS ketika sudah terbang berpindah tempat hingga koordinat yang dituju tercapai. Setelah dititik akhir tujuan *waypoint* akan dilakukan pendaratan atau *return to launch*. Setelah itu alur pun selesai.

.

4. Komunikasi Antara *Drone* dengan GCS

Untuk melakukan komunikasi antara *drone* dengan GCS digunakan modul telemerti. Modul telemetri yang digunakan adalah tipe 3DR Radio Telemetry dengan jangkauan yang dimiliki dapat hingga 1 Km. Modul ini akan terpasang kepada masing – masing perangkat baik laptop sebagai GCS maupun pada *drone* keduanya akan menjadi *transmitter* dan juga *receiver* secara bergantian. Karena perintah akan dikirim melalui laptop kemudian diterima oleh *drone*. Dan juga *drone* akan selalu mengirimkan informasi mengenai dirinya baik itu lokasi, kestabilan, ketinggian, daya baterai, kecepatan gerak, sampai dengan jika *drone* mengalami kendala.

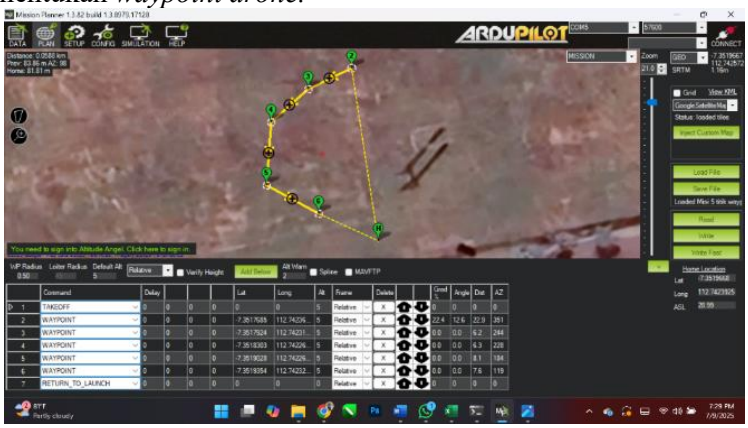
Proses koneksi pada Mission Planner dapat terlihat seperti gambar 5, ditandai dengan warna hijau pada kotak kanan. Kotak kanan atas menunjukkan bahwa laptop dan *drone* sudah terkoneksi atau belum. Untuk melakukan koneksi sebelumnya diperlukan penyesuaian. Pertama terlebih dahulu ditentukan *port usb* berapa yang digunakan dengan terpasangnya modul telemetri. Pada contoh gambar diatas port yang terpakai adalah COM 5. Kemudian ditentukan *baudrate* yang digunakan, karena dalam penelitian ini menggunakan modul telemetri sehingga *baudrate* yang digunakan adalah 57600 untuk dapat berkomunikasi dengan *Pixhawk* pada *drone*. Jika setingan parameter sudah dilakukan koneksi dapat dilakukan dengan menekan kotak ”*connect”* kemudian tunggu hingga proses koneksi selesai.

Perancangan Misi dan Navigasi

Setelah *hardware* terpasang dan sistem sudah siap, langkah selanjutnya adalah perancangan misi penerbangan yang mencakup:

a. ***Waypoint Navigation*** yaitu menentukan titik-titik *waypoint* yang harus dilalui oleh *drone* secara otomatis(Saputro et al., 2021).
b. ***Auto Takeoff dan Landing*** yaitu menerapkan fitur *auto takeoff* dan *auto landing* untuk memastikan penerbangan yang aman dan stabil(Achmad et al., 2025).
c. ***Return to Launch (RTL)*** yaitu pemrograman sistem agar *drone* dapat kembali ke titik awal secara otomatis jika terjadi kehilangan sinyal atau pada akhir misi(Tamtomi et al., 2016).

Misi penerbangan ini akan diprogram dan dipantau menggunakan Mission Planner, dengan ArduPilot yang menjalankan misi secara *real-time*. Berikut merupakan tampilan pada Mission Planner untuk menentukan *waypoint drone.*



Gambar 5. Tampilan pembuatan misi di Mission Planner

Menentukan titik dapat dilakukan dengan menandai (titik) langsung pada map yang tertera pada tampilan mission planner seperti Gambar 5. Pada titik pertama yaitu *home base* dipilih perintah *“take off”* pada kotak *command* dan pilih perintah “*waypoint*” pada titik – titik selanjutnya. Untuk titik terakhir perintah yang dipilih adalah *“land”* guna memerintahkan *drone* untuk melakukan pendararatan atau juga bisa memilih perintah *“return to launch”* untuk melakukan pendaratan kembali pada titik *take off*. Setelah penentuan *waypoint* dan melakukan pengiriman misi ke *Pixhawk* dengan menekan *button* “*write*”. Rute plan juga dapat disimpan jika rute akan dijalankan kembali dengan *button* ”*save file*”. Dan memasukkan kembali rute yang telah disimpan dengan *button* “*load file”*.

Gambar 6. Tampilan pada Mission Planner

Pada Gambar 6, tampilan kanan memberikan gambaran *drone* pada peta dimana lokasi *drone* berada dan juga pergerakannya secara *real-time.* Kemudian pada sisi kiri atas menampilkan keseimbangan *drone* seperti tampilan pesawat atau helikopter, informasi jika *drone* mengalami kendala dan kondisi baterai. Pada sisi kiri bawah memiliki beberapa opsi yang ingin dipilih diantaranya untuk menampilkan ketinggian, jarak, dan kecepatan ada pada tab *"Quick"*. Sedangkan untuk melakukan kendali *autonomous* pada tab *"Actions"*. Untuk melakukan perintah *autonomous* adalah dengan melakukan pemilihan perintah "Mission Start" (lingkaran merah). Dilanjutkan dengan menekan tombol button "Arm/Diasrm" jika *drone* masih dalam kondisi "Diasarmed" (bertulisakan teks merah) dilanjutkan dengan menekan *button* "Do Action" dan pilih "Yes". Kecepatan dan ketinggian pada *drone* juga dapat ditentukan saat melakukan mode *autonomous.*

III. HASIL DAN PEMBAHASAN

1. Pengujian Autonomous dengan Jarak Tempuh

Untuk mengetahui *drone* dapat dijalankan secara *autonomous* diperlukan beberapa uji coba. Uji coba pertama adalah dengan melakukan uji jarak tempuh dikarenakan sistem *autonomous* sangat berpengaruh terhadap komunikasi *drone* dangan GCS. Jarak yang diambil untuk melakukan uji adalah 10m, 20m, 30m, 40m, dan 50m. Jarak ini diukur melalui tampilan yang tertera pada Mission Planner kemudian dibuktikan dengan alat ukur meteran gulung. Titik awal pengukuran diambil dari titik *home base*. Uji pertama adalah pengujian jarak penerbangan *drone* secara *autonomous* pada jarak 10 m. Tampilan rute 10 m pada Mission Planner tertampil pada Gambar 7. Penentuan jarak 10m dapat dilihat dari kotak dengan keterangan "dist" yang berarti *distance* (jarak).

Gambar 7. *Plan rute* pada tampilan *Mission Planner* dengan jarak 10m

Kemudian dilanjutkan dengan uji coba terbang pada *drone* secara *autonomous* dan dilakukan pendaratan dititik tujuan. Untuk membuktikan bahwa jarak yang ditempuh sudah sesuai maka jarak tempuh juga diukur dengan meteran secara manual, sehingga bisa dibuktikan bahwa jarak yang ditempuh sudah benar. Pada tahap pengujian terbang ini dilakukan sebanyak 10x untuk memastikan bahwa sistem *autonomous* pada *drone* untuk melakukan misi dapat dipastikan keberhasilannya. Hasil yang didapat dari tes dengan jarak 10 m tertera pada tabel berikut.

Tabel 1. Uji *drone* dengan jarak 10m

Test Autonomous Jarak 10m			
No.	Uji Coba	Terbang	Mendarat
1	Percobaan ke-1	Aman (Tidak Jatuh)	Berhasil
2	Percobaan ke-2	Aman (Tidak Jatuh)	Berhasil
3	Percobaan ke-3	Aman (Tidak Jatuh)	Berhasil
4	Percobaan ke-4	Aman (Tidak Jatuh)	Berhasil
5	Percobaan ke-5	Aman (Tidak Jatuh)	Berhasil
6	Percobaan ke-6	Aman (Tidak Jatuh)	Berhasil
7	Percobaan ke-7	Aman (Tidak Jatuh)	Berhasil
8	Percobaan ke-8	Aman (Tidak Jatuh)	Berhasil
9	Percobaan ke-9	Aman (Tidak Jatuh)	Berhasil
10	Percobaan ke-10	Aman (Tidak Jatuh)	Berhasil

Setelah dilakukakan pengujian, *drone* dapat melakukan misi terbang dengan aman tanpa terjatuh hingga titik *waypoint* yang telah ditentukan lalu berhasil melakukan pendaratan dengan stabil. Dengan langkah yang sama uji coba dilanjutkan dengan jarak 20 m dan 50 m, hasil percobaan terlihat pada Tabel 2.

Tabel 2. Uji *drone* dengan jarak 20 m dan 50 m

No.	Uji Coba	Test Autonomous Jarak 20 m		Jarak 50 m	
		Terbang	Mendarat	Terbang	Mendarat
1	Percobaan ke-1	Aman (Tidak Jatuh)	Berhasil	Aman (Tidak Jatuh)	Berhasil
2	Percobaan ke-2	Aman (Tidak Jatuh)	Berhasil	Aman (Tidak Jatuh)	Berhasil
3	Percobaan ke-3	Aman (Tidak Jatuh)	Berhasil	Aman (Tidak Jatuh)	Berhasil
4	Percobaan ke-4	Aman (Tidak Jatuh)	Berhasil	Aman (Tidak Jatuh)	Berhasil
5	Percobaan ke-5	Aman (Tidak Jatuh)	Berhasil	Aman (Tidak Jatuh)	Berhasil
6	Percobaan ke-6	Aman (Tidak Jatuh)	Berhasil	Terjatuh	Gagal
7	Percobaan ke-7	Aman (Tidak Jatuh)	Berhasil	Terjatuh	Gagal
8	Percobaan ke-8	Aman (Tidak Jatuh)	Berhasil	Aman (Tidak Jatuh)	Berhasil
9	Percobaan ke-9	Aman (Tidak Jatuh)	Berhasil	Aman (Tidak Jatuh)	Berhasil
10	Percobaan ke-10	Aman (Tidak Jatuh)	Berhasil	Aman (Tidak Jatuh)	Berhasil

Pada jarak 20m percobaan dapat berjalan lancer tidak ada kendala, tetapi terdapat kendala dimana *drone* mengalami kegagalan mendarat dan terjatuh pada jarak 50 m. Hal ini dikarenakan power supply dari baterai yang mengalami penurunan, sehingga uji coba harus dilanjutkan setelah baterai diisi (*charge*) kembali, kemudian setelah baterai terisi penuh *drone* dapat berjalan dengan baik, hal ini menjadi catatan untuk pemilihan baterai yang sesui dengan kebutuhan.

2. Uji Coba Autonomous dengan Jarak Terjauh dan Maximum Kecepatan

Setelah dilakukan uji coba terbang secara autonomous dengan beberapa jarak, penullis melakukan pengujian pada seberapa jauh jarak yang dapat ditempuh oleh *drone* jika secara autonomous. Jarak diambil dilakukan di sebuah lapangan terbuka dengan jarak mencapai batas terjauh dari lapangan tersebut. Kecepatan juga menggunakan beberapa variasi untuk melihat batas kecepatan dari *drone*. Kemudian dari percobaan yang telah dilakukan didapatkan hasil tabel berikut.

Tabel 3. Uji drone jarak terjauh dengan beberapa kecepatan

Test Autonomous Jarak Terjauh dengan Beberapa Kecepatan									
Uji Coba	Jarak 160 m (*Waypoint* 1)			Jarak 58 m (*Waypoint* 2)			Jarak 152 m (*Waypoint* 3)		
	Kecepatan			Kecepatan			Kecepatan		
	25 m/s	50 m/s	100 m/s	25 m/s	50 m/s	100 m/s	25 m/s	50 m/s	100 m/s
1	Stabil	Stabil	Sedikit Penurunan	Stabil	Stabil	Sedikit Penurunan	Stabil	Stabil	Sedikit Penurunan
2	Stabil	Stabil	Sedikit Penurunan	Stabil	Stabil	Sedikit Penurunan	Stabil	Stabil	Sedikit Penurunan
3	Stabil	Stabil	Sedikit Penurunan	Stabil	Stabil	Sedikit Penurunan	Stabil	Stabil	Sedikit Penurunan
4	Stabil	Stabil	Sedikit Penurunan	Stabil	Stabil	Sedikit Penurunan	Stabil	Stabil	Sedikit Penurunan
5	Stabil	Stabil	Sedikit Penurunan	Stabil	Stabil	Sedikit Penurunan	Stabil	Stabil	Sedikit Penurunan
6	Stabil	Stabil	Sedikit Penurunan	Stabil	Stabil	Sedikit Penurunan	Stabil	Stabil	Sedikit Penurunan
7	Stabil	Stabil	Sedikit Penurunan	Stabil	Stabil	Sedikit Penurunan	Stabil	Stabil	Sedikit Penurunan
8	Stabil	Stabil	Sedikit Penurunan	Stabil	Stabil	Sedikit Penurunan	Stabil	Stabil	Sedikit Penurunan
9	Stabil	Stabil	Sedikit Penurunan	Stabil	Stabil	Sedikit Penurunan	Stabil	Stabil	Sedikit Penurunan
10	Stabil	Stabil	Sedikit Penurunan	Stabil	Stabil	Sedikit Penurunan	Stabil	Stabil	Sedikit Penurunan

Pada saat proses terbang dari titik peluncuran menuju *waypoint* pertama dan pada saat dari *waypoint* kedua menuju *waypoint* ketiga mengalami penurunan. Penurunan yang dimaksud disini adalah penurunan ketinggian. Hal tersebut dikarenakan tingkat kecepatan mempengaruhi posisi derajat terbang dari *drone*. Namun penurunan ketinggian yang terjadi segera dikoreksi oleh *drone* ketika mencapai titik tujuan.sebelum melakukan perintah selanjutnya atau menuju *waypoint* selanjutnya.

3. Uji Coba Autonomous dengan *Maximum* Ketinggian

Untuk menentukan kemampuan seberapa tinggi *drone* dapat terbang uji coba dilakukan pada beberapa tingkat ketinggian. Ketinggian diuji pada ketinggian 5 m, 10 m, dan 15 m. Kemudian didapatkan hasil pengujian sebagai berikut.

Tabel 4. Uji coba drone secara autonomous dengan masing-masing ketinggian

Test Autonomous Ketinggian 5 m				
No.	Uji Coba	Cara Terbang		
		5m	10m	15m
1	Percobaan ke-1	Stabil	Stabil	Stabil
2	Percobaan ke-2	Stabil	Stabil	Sedikit terguncang
3	Percobaan ke-3	Stabil	Stabil	Stabil

4	Percobaan ke-4	Stabil	Stabil	Stabil
5	Percobaan ke-5	Stabil	Stabil	Sedikit terguncang
6	Percobaan ke-6	Stabil	Stabil	Sedikit terguncang
7	Percobaan ke-7	Stabil	Stabil	Stabil
8	Percobaan ke-8	Stabil	Stabil	Sedikit terguncang
9	Percobaan ke-9	Stabil	Stabil	Sedikit terguncang
10	Percobaan ke-10	Stabil	Stabil	Stabil

Setelah dilakukan uji ketinggian didapatkan pada ketinggian 15 m *drone* mengalami guncangan ketika terbang menuju *waypoint* ataupun juga melakukan *return to launch*. Efek guncangan ini disebabkan oleh hembusan tekanan angin yang berbeda disetiap ketinggian.

4. Uji Coba Autonomous dengan beberapa titik *Waypoint*

Setelah dilakukan uji coba jarak kemampuan *drone* bisa terbang secara autonomous sebelumnya hingga 50 m dilakukan uji coba *drone* dengan waktu tempuh yang lebih panjang dengan menggunakan beberapa titik *waypoint*. Jumlah tujuan titik *waypoint* saya bagi menjadi 2 macam yaitu 3 titik *waypoint* dan 5 titik *waypoint* dan masing - masing diakhiri dengan *drone* melakukan pendaratan ditempat peluncuran. Masing – masing pengujian dilakukan sebanyak 10x percobaan. Berikut rute plan yang diberikan kepada *drone* untuk dijalankan.

Gambar 8. Plan rute pada tampilan Mission Planner dengan 3 titik waypoint

Titik 2, 3, dan 4 pada gambar diatas merupakan titik *wayoint* tujuan untuk dijalankan oleh *drone* yang diinginkan penulis sebagai uji coba *drone* dengan misi 3 titik. Kemudian setelah *drone* menjalankan misi menuju masing – masing misi *drone* diperintahkan untuk kembali ke tempat peluncuran. Hasil dari uji coba *drone* secara autonomous dengan 3 titik tertera pada Tabel 6.

Tabel 5. Uji coba drone autonomous dengan 3 titik waypoint

Test Autonomous 3 Titik *Waypoint*					
No.	Uji Coba	Titik Pertama	Titik Kedua	Titik Ketiga	Mendarat
1	Percobaan ke-1	Aman (Tidak Jatuh)	Aman (Tidak Jatuh)	Aman (Tidak Jatuh)	Berhasil
2	Percobaan ke-2	Aman (Tidak Jatuh)	Aman (Tidak Jatuh)	Aman (Tidak Jatuh)	Berhasil
3	Percobaan ke-3	Aman (Tidak Jatuh)	Aman (Tidak Jatuh)	Aman (Tidak Jatuh)	Berhasil
4	Percobaan ke-4	Aman (Tidak Jatuh)	Aman (Tidak Jatuh)	Aman (Tidak Jatuh)	Berhasil
5	Percobaan ke-5	Aman (Tidak Jatuh)	Aman (Tidak Jatuh)	Aman (Tidak Jatuh)	Berhasil
6	Percobaan ke-6	Aman (Tidak Jatuh)	Aman (Tidak	Aman (Tidak	Berhasil

			Jatuh)	Jatuh)	
7	Percobaan ke-7	Aman (Tidak Jatuh)	Aman (Tidak Jatuh)	Aman (Tidak Jatuh)	Berhasil
8	Percobaan ke-8	Aman (Tidak Jatuh)	Aman (Tidak Jatuh)	Aman (Tidak Jatuh)	Berhasil
9	Percobaan ke-9	Aman (Tidak Jatuh)	Aman (Tidak Jatuh)	Aman (Tidak Jatuh)	Berhasil
10	Percobaan ke-10	Aman (Tidak Jatuh)	Aman (Tidak Jatuh)	Aman (Tidak Jatuh)	Berhasil

Hasil menunjukkan bahwa *drone* dapat melakukan misi dengan baik sesuai dengan apa yang diperintahkan dan kembali lagi mendarat dengan sempurna. Untuk menguji kemampuan *drone* dalam hal memproses perintah uji selanjutnya diberikan 2 titik tambahan menjadi 5 titik *waypoint* kemudian *drone* kembali ke tempat peluncuran. Berikut rute plan yang diberikan kepada *drone* untuk dijalankan. Hasil dari uji coba *drone* secara autonomous dengan 5 titik tertera pada Tabel 7 berikut ini.

Tabel 6. Uji coba drone autonomous dengan 5 titik waypoint

	Test Autonomous 5 Titik *Waypoint*					
Uji Coba	Titik Pertama	Titik Kedua	Titik Ketiga	Titik Keempat	Titik Kelima	Mendarat
1	Aman (Tidak Jatuh)	Aman (Tidak Jatuh)	Aman (Tidak Jatuh)	Aman (Tidak Jatuh)	Aman (Tidak Jatuh)	Berhasil
2	Aman (Tidak Jatuh)	Aman (Tidak Jatuh)	Aman (Tidak Jatuh)	Aman (Tidak Jatuh)	Aman (Tidak Jatuh)	Berhasil
3	Aman (Tidak Jatuh)	Aman (Tidak Jatuh)	Aman (Tidak Jatuh)	Aman (Tidak Jatuh)	Aman (Tidak Jatuh)	Berhasil
4	Aman (Tidak Jatuh)	Aman (Tidak Jatuh)	Aman (Tidak Jatuh)	Aman (Tidak Jatuh)	Aman (Tidak Jatuh)	Berhasil
5	Aman (Tidak Jatuh)	Aman (Tidak Jatuh)	Aman (Tidak Jatuh)	Aman (Tidak Jatuh)	Aman (Tidak Jatuh)	Berhasil
6	Aman (Tidak Jatuh)	Aman (Tidak Jatuh)	Aman (Tidak Jatuh)	Aman (Tidak Jatuh)	Aman (Tidak Jatuh)	Berhasil
7	Aman (Tidak Jatuh)	Aman (Tidak Jatuh)	Aman (Tidak Jatuh)	Aman (Tidak Jatuh)	Aman (Tidak Jatuh)	Berhasil
8	Aman (Tidak Jatuh)	Aman (Tidak Jatuh)	Aman (Tidak Jatuh)	Aman (Tidak Jatuh)	Aman (Tidak Jatuh)	Berhasil
9	Aman (Tidak Jatuh)	Aman (Tidak Jatuh)	Aman (Tidak Jatuh)	Aman (Tidak Jatuh)	Aman (Tidak Jatuh)	Berhasil
10	Aman (Tidak Jatuh)	Aman (Tidak Jatuh)	Aman (Tidak Jatuh)	Aman (Tidak Jatuh)	Aman (Tidak Jatuh)	Berhasil

Dari uji coba terakhir dapat dinyatakan *drone* sudah dapat dikendalikan secara autonomous tanpa menggunakan remot. *Drone* dengan aman melakukan misi terbang ketempat titik – titik *waypoint* dengan baik. Hanya saja pengaruh keakuratan dari titik GPS yang mempengaruhi juga ketepatan titik yang diminta untuk dilakukan oleh *drone*. Namun, *error* pada keakuratan titik *waypoint* yang diminta masih dalam batas yang wajar yaitu masih sekitar radius kurang lebih 50 cm.

IV.SIMPULAN

Berdasarkan hasil uji coba yang telah dilakukan, sistem *drone* quadcopter yang dikendalikan secara autonomous menggunakan flight controller *Pixhawk* dan software ArduPilot menunjukkan performa yang baik dan stabil. *Drone* berhasil melakukan proses take-off, navigasi *waypoint*, serta landing secara otomatis tanpa mengalami gangguan teknis atau jatuh. Keberhasilan dan kestabilan selama penerbangan juga terjaga dengan baik pada ketinggian hingga 10 m dengan kecepatan 50 m/s. Namun pada uji coba pada ketinggian 15 m *drone* mengalami guncangan dikarenakan

.

hembusan angin yang memiliki tekanan lebih tinggi pada ketinggian tersebut. Nilai kecepatan semakin tinggi yang telah dilakukan dengan nilai 100 m/s juga membuat *drone* mengalami penurunan ketinggian dikarenakan sudut derajat *drone*. Dari uji coba ini dapat disimpukan bahwa *drone* masih harus dikembangkan kembali agar sistem autonomous dapat mecapai batas lebih maksimal lagi. Dengan nilai roll Kp = 0,135; Ki = 0,135; Kd = 0,0036, nilai pitch Kp = 0,135; Ki = 0,135; Kd = 0,0036; dan nilai yaw Kp = 0,180; Ki = 0,018; Kd = 0 yang digunakan sekarang masih membuat *drone* mengalami gangguan pada batas yg telah disebutkan diatas.

REFERENSI

Hayat, A., & Morgado-Dias, F. (2022). *Deep learning* -based automatic safety helmet detection system for construction safety. *Applied Sciences*, *12*(16), 8268.

Hidayat, M. S., Pagiling, L., & Nur, M. N. A. (2019). Perancangan sistem pengepakan otomatis berbasis arduino uno menggunakan sensor jarak infra red. *J. Fokus Elektroda Energi List. Telekomun. Komputer, Elektron. Dan Kendali*, *4*(1), 1–8.

Hudha, M., Supriyati, E., & Listyorini, T. (2022). Analisis Sentimen Pengguna Youtube Terhadap Tayangan# Matanajwamenantiterawan Dengan Metode Naïve Bayes Classifier. *JIKO (Jurnal Inform. Dan Komputer)*, *5*(1), 1–6.

Hutauruk, J. S. W., Matulatan, T., & Hayaty, N. (2020). Deteksi kendaraan secara real time menggunakan metode YOLO berbasis android. *Jurnal Sustainable: Jurnal Hasil Penelitian Dan Industri Terapan*, *9*(1), 8–14.

Nurdiansyah, A. (2020). *Peringkasan Teks Dokumen Menggunakan Metode Simple Additive Weighting (Saw)*. Universitas Komputer Indonesia.

Putra, P. Y., Arifianto, A. S., Fitri, Z. E., & Puspitasari, T. D. (2023). Deteksi Kendaraan Truk pada Video Menggunakan Metode Tiny-YOLO v4. *Jurnal Informatika Polinema*, *9*(2), 215–222.

Rahma, L., Syaputra, H., Mirza, A. H., & Purnamasari, S. D. (2021). Objek Deteksi Makanan Khas Palembang Menggunakan Algoritma YOLO (You Only Look Once). *Jurnal Nasional Ilmu Komputer*, *2*(3), 213–232.

Rofii, F., Priyandoko, G., Fanani, M. I., & Suraji, A. (2021). Vehicle Counting Accuracy Improvement By Identity Sequences Detection Based on Yolov4 Deep Neural Networks. *Teknik*, *42*(2), 169–177.

Salamah, I., Said, M. R. A., & Soim, S. (2022). Perancangan Alat Identifikasi Wajah Dengan Algoritma You Only Look Once (YOLO) Untuk Presensi Mahasiswa. *J. Media Inform. Budidarma*, *6*(3), 1492.

Soeltanong, M. B., & Sasongko, C. (2021). Perencanaan produksi dan pengendalian persediaan pada perusahaan manufaktur. *JRAP (Jurnal Riset Akuntansi Dan Perpajakan)*, *8*(1), 14–27.

Tawakkal, M. A. (2024). *Rancang Bangun Aplikasi Penghitungan Okupansi Manusia Dengan CCTV Menggunakan Object Detection Model YOLOv8*. Universitas Hasanuddin Makassar.

UTOMO, M. S. (2023). *Deteksi Jumlah Manusia Menggunakan Metode YOLOv4 Dalam Suatu Gedung (Studi Kasus: Lab Elektro Unissula)*. Universitas Islam Sultan Agung.

Widoretno, S., & Mahardika, A. M. A. F. (2024). Conveyor Belt dan Alat Penghitung Otomatis Berbasis Arduino Nano Menggunakan Sensor Inframerah pada Produksi Roti Tawar. *Jurnal Qua Teknika*, *14*(1), 87–99.

Yanto, Y., Aziz, F., & Irmawati, I. (2023). YOLO-V8 peningkatan algoritma untuk deteksi pemakaian masker wajah. *JATI (Jurnal Mahasiswa Teknik Informatika)*, *7*(3), 1437–1444.

Yasen, N. M., Rifka, S., Vitria, R., & Yulindon, Y. (2023). Pemanfaatan YOLO untuk deteksi hama dan penyakit pada daun cabai menggunakan metode *deep learning* . *Elektron: Jurnal Ilmiah*, 63–71.