

# Kaji Pengolahan Citra Quadcopter v10 untuk Misi Terbang *Object Detection* ArUco Marker

Iqlima Khairunnisa<sup>1</sup>, Budi Hartono<sup>2\*</sup>

<sup>1,2</sup>Jurusan Teknik Mesin, Politeknik Negeri Bandung, Bandung 40012

<sup>1</sup>E-mail : iqlima.khairunnisa.aer22@polban.ac.id

<sup>2\*</sup>E-mail : buhar@polban.ac.id

## ABSTRAK

Teknologi *object detection* menjadi komponen penting dalam sistem navigasi dan pendaratan presisi pada wahana udara tak berawak (*Unmanned Aerial Vehicle*). Penelitian ini mengembangkan sistem pengolahan citra pada Quadcopter v10 untuk menjalankan misi terbang secara otonom dengan mendeteksi ArUco marker sebagai target pendaratan. *Marker* ini dipilih karena kemampuannya dalam memberikan identifikasi *visual* yang stabil berbasis pola biner. Proses perancangan dimulai dengan perakitan quadcopter menggunakan konfigurasi rangka X, *flight controller* Pixhawk 2.4.8, serta Raspberry Pi 4 Model B yang terintegrasi dengan kamera Logitech C270 sebagai sistem pemroses citra. Kalibrasi kamera dilakukan menggunakan metode Zhang untuk mendapatkan parameter intrinsik dan koreksi distorsi. Sistem deteksi *marker* diimplementasikan secara *real-time* dengan pustaka OpenCV dan ArUco, sementara estimasi pose dihitung menggunakan fungsi `cv2.aruco.estimatePoseSingleMarkers()` yang menghasilkan vektor rotasi dan translasi *marker* terhadap kamera. Hasil pengujian menunjukkan bahwa sistem mampu mendeteksi marker ID 23 secara konsisten, serta menampilkan estimasi posisi dalam bentuk visualisasi sumbu koordinat 3D. Dengan demikian, sistem ini menunjukkan efektivitas penerapan teknologi *object detection* berbasis ArUco marker dalam mendukung proses identifikasi posisi untuk kebutuhan navigasi *visual* UAV.

### Kata Kunci

object detection, ArUco marker, quadcopter, estimasi pose

*Object detection technology plays a crucial role in navigation and precision landing systems for Unmanned Aerial Vehicles (UAVs). This project developed a computer vision system on the Quadcopter V10 to perform autonomous flight missions by detecting ArUco markers as landing targets. ArUco markers were chosen due to their binary pattern, which allows for stable and reliable visual identification. The system design involved assembling a quadcopter with an X-frame configuration, using a Pixhawk 2.4.8 flight controller, and integrating a Raspberry Pi 4 Model B with a Logitech C270 camera for visual data processing. Camera calibration was performed using the Zhang method to obtain intrinsic parameters and correct lens distortion. Marker detection was implemented in real-time using the OpenCV and ArUco libraries, while pose estimation was carried out using the cv2.aruco.estimatePoseSingleMarkers() function to generate rotation and translation vectors relative to the camera. Test results showed that the system was able to consistently detect marker ID 23 and display the estimated pose as a 3D coordinate axis visualization. This demonstrates the effectiveness of ArUco marker-based object detection technology in supporting visual navigation systems for UAVs.*

### Keywords

object detection, ArUco marker, quadcopter, pose estimation

## 1. PENDAHULUAN

*Unmanned Aerial Vehicles* (UAV) atau kendaraan udara tak berawak semakin marak dalam berbagai bidang seperti pengawasan, pemetaan, dan pengiriman barang. Salah satu fokus utama dalam penggunaan UAV adalah sistem pendaratan yang presisi, yang sangat penting untuk mencegah kerusakan dan kecelakaan. Sistem deteksi UAV biasanya menggunakan proses *multistage*, termasuk langkah-langkah seperti *thresholding* gambar, penghapusan latar belakang, dan pelacakan objek.

Lebedev *et al.* (2020) menyusun jurnal yang berjudul "Accurate Autonomous UAV Landing Using Vision-Based Detection of ArUco-Marker". Penelitian ini membahas

solusi untuk masalah pendaratan presisi secara *autonomous* UAV di titik target. Hasil dari penelitian ini mengusulkan kombinasi dua algoritma untuk pencarian objek pada gambar dan pendaratan yang akurat pada *marker*. Sistem ini mencakup unit untuk menyesuaikan posisi UAV relatif terhadap *marker* untuk meningkatkan akurasi pendaratan.(1)

Utami (2024) menyusun Tugas Akhir dengan judul "Quadcopter V9: Kaji Pengolahan Citra Untuk Misi Terbang *Object detection*". Pada sistem pengolahan citra pada ketinggian 3 m, quadcopter ini mampu mendeteksi objek lingkaran merah dengan ukuran 40 cm. Namun terdapat kendala pada pengujian ini, sistem pengolahan citra tidak dapat mendeteksi objek dengan diameter 60 cm pada ketinggian 2, 3, dan 4 m. Selain itu pada objek dengan diameter 40 cm, hanya terdeteksi pada ketinggian 3 m.(2)(3)

ArUco marker sendiri merupakan *marker visual* berbentuk kotak yang terdiri dari pola biner hitam-putih yang tersusun dalam grid persegi. Setiap *marker* memiliki identitas unik yang dikodekan dalam pola tersebut, sehingga dapat dikenali secara otomatis oleh sistem pengolahan citra. Ciri khas dari ArUco marker adalah kemampuannya untuk dideteksi secara *real-time* bahkan dalam kondisi rotasi dan skala yang berbeda.(4)

Pada penelitian ini, quadcopter akan melakukan misi terbang *object detection* terhadap ArUco marker. Quadcopter menggunakan sistem pengolahan citra untuk mendeteksi objek, dan mendarat di atasnya. Quadcopter akan melakukan *take off* dan mengaktifkan sistem pengolahan citra. sistem deteksi ArUco dapat berjalan secara *real-time*, bahkan pada perangkat dengan kemampuan komputasi yang terbatas, seperti *single-board computer* (misalnya Raspberry Pi). Sistem hanya perlu mengidentifikasi pola kotak hitam-putih dan mencocokkannya dengan *database ID marker*. ArUco marker juga mendukung estimasi pose tiga dimensi (3D), yang mencakup informasi posisi (translasi) dan orientasi (rotasi) relatif terhadap kamera. Kemampuan ini sangat penting dalam sistem navigasi, karena memungkinkan quadcopter menyesuaikan posisi dan arah secara presisi saat mendekati target atau melakukan pendaratan tepat di atas *marker*.(5)

## 2. LANDASAN TEORI

### 2.1 Quadcopter

Quadcopter adalah *drone multirotor* yang dilengkapi dengan empat motor untuk penerbangan. Dua motor berputar searah jarum jam dan dua lainnya berputar berlawanan arah jarum jam. Dengan keempat rotornya, quadcopter dapat bermanuver ke segala arah dengan sangat fleksibel. Kemampuan ini dimanfaatkan manusia untuk menyederhanakan pekerjaan di lingkungan berbahaya, seperti pemantauan kebakaran hutan.(6)

### 2.2 Pengolahan Citra

Pengolahan citra merupakan suatu proses di mana citra digunakan sebagai *input* untuk menghasilkan citra *output* sesuai dengan yang diinginkan. Walaupun mata dapat melihat langsung objek atau pemandangan tanpa alat bantu, rekaman yang terjadi dalam otak bersifat relatif dan subjektif. Dapat dikatakan relatif, karena hanya mampu mengingat bagian-bagian yang lebih terang dan lebih gelap, sedangkan sifat subjektif karena dipengaruhi oleh pengalaman dan harapan yang berbeda-beda antara individu. Oleh karena itu, diperlukan alat untuk menangkap dan menyimpan citra secara objektif, yaitu yang dapat disajikan berdasarkan sistem ukuran yang standar.(7)

### 2.3 Raspberry Pi

Raspberry Pi adalah komputer papan tunggal yang diperkenalkan pada tahun 2012 dan banyak digunakan dalam pengembangan sistem *Internet of Things* (IoT) berkat fleksibilitasnya untuk menghubungkan berbagai antarmuka sensor, aktuator, dan modul komunikasi. Fleksibilitas ini

menjadikan Raspberry Pi ideal digunakan sebagai pusat kendali dalam sistem quadcopter berbasis IoT, dimana data sensor dan aktuator dapat diproses secara *real-time* dan modul komunikasi memungkinkan kontrol jarak jauh yang efisien.(8)

### 2.4 ArUco Marker

ArUco marker merupakan penanda buatan berbentuk persegi yang terdiri dari tepi hitam lebar dan matriks *biner* di dalamnya yang menentukan identifikasinya (id). Tepi hitam memfasilitasi deteksi cepat dalam gambar, dan pengkodean *biner* memungkinkan identifikasi serta penerapan teknik deteksi dan koreksi kesalahan. Ukuran penanda menentukan ukuran matriks *internal*. Misalnya, penanda berukuran 4x4 terdiri dari 16 *bit*. Modul ArUco mencakup beberapa kamus yang telah ditentukan sebelumnya, mencakup berbagai ukuran kamus dan ukuran penanda yang berbeda.(9)

## 3. METODE PENELITIAN

### 3.1 Perakitan Quadcopter

Perakitan Quadcopter v10 diawali dengan pemilihan komponen perangkat keras yang disesuaikan dengan kebutuhan sistem untuk menjalankan misi terbang *object detection*. Struktur utama menggunakan frame DJI F450, dengan sistem kendali penerbangan berbasis *flight controller* Pixhawk PX4 2.4.8 32-bit. Untuk mendukung kestabilan dan navigasi, digunakan sensor jarak LiDAR TF Benewake serta modul GPS M8N. Sistem tenaga disuplai oleh baterai Li-ion 4 cell berkapasitas 5000 mAh, yang distribusinya diatur oleh *power module* 3DR Ardupilot dengan spesifikasi tegangan 6–18 V dan arus maksimum 60 A.(10)

Penggerak utama terdiri dari motor Sunnysky X2216 1400KV yang dikendalikan oleh ESC Diatone Mamba 50A tipe 4-in-1, dengan *propeller* berjenis 1245 CW dan CCW berbahan nilon fiber. Sistem komunikasi dilakukan menggunakan radio telemetri RFD 900X, sementara pengendalian manual tetap dimungkinkan melalui *receiver* dari RadioLink. Untuk menjaga kualitas daya dan pengisian ulang, digunakan *charger* baterai IMAX B6. Selain itu, *damper* khusus dipasang untuk meredam getaran pada *flight controller* guna meningkatkan akurasi sensor.

Seluruh komponen dirakit pada *frame* dengan konfigurasi X, yang umum digunakan karena memberikan kestabilan aerodinamis serta distribusi beban yang seimbang pada masing-masing lengan motor. Perancangan ini menjadi dasar sistem navigasi dan integrasi lanjutan dengan unit pengolahan citra.

### 3.2 Perancangan Sistem Pengolahan Citra dan *Autonomous*

Pada proses ini, maka quadcopter sudah berhasil terbang sesuai dengan kontrol yang dicoba. Tahap selanjutnya yaitu menerapkan sistem pemrograman untuk menjalankan misi secara *autonomous*. Pada tahap ini, sistem dirancang agar quadcopter dapat melakukan deteksi objek secara mandiri dan menyesuaikan pergerakan berdasarkan hasil pengolahan

citra yang diterima dari kamera. Bahasa pemrograman Python 3.11 dengan ArUco *library* digunakan dalam sistem pengolahan citra dengan misi terbang ini.(11)

### 3.3 Perakitan Perangkat Keras

Sebelum melakukan misi terbang *object detection*, quadcopter memerlukan perancangan sistem pengolahan citra. Sistem pengolahan citra memerlukan komputer untuk mengolah data yang akan ditangkap oleh sensor. Data *input* citra yang ditangkap kamera akan diolah oleh Raspberry Pi 4 Model B. Gambar 1 merupakan skematik rangkaian sistem pengolahan citra.



Gambar 1. Skematik Rangkaian Pengolahan Citra

Raspberry Pi akan mengelola data tersebut sebelum diteruskan ke *flight controller* untuk koordinasi navigasi berdasarkan deteksi objek. *flight controller* terhubung ke GCS (*Ground Control Station*) melalui modul komunikasi, memungkinkan operator memantau dan mengontrol misi secara langsung. Seluruh sistem mendapat catu daya dari baterai Li-Po yang dihubungkan melalui konektor khusus. Sistem ini juga dilengkapi ESC (*Electronic Speed Controller*) untuk mengatur kecepatan motor berdasarkan instruksi dari *flight controller*. Pemrosesan citra secara onboard menggunakan Raspberry Pi dinilai efektif untuk mendukung misi UAV ringan karena konsumsi daya yang rendah serta kemampuannya menjalankan algoritma deteksi berbasis Python dan OpenCV. Oleh karena itu, integrasi antara perangkat keras dan perangkat lunak ini sangat penting untuk menjamin keberhasilan misi otonom berbasis visi komputer.

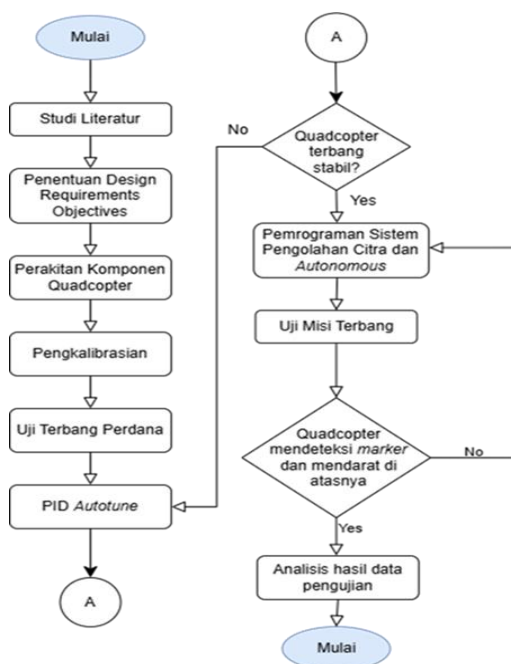
### 3.4 Kalibrasi Kamera

Kalibrasi kamera merupakan tahapan penting dalam sistem pengolahan citra berbasis estimasi pose, khususnya untuk mendeteksi dan menentukan posisi objek secara akurat di dunia nyata. Tujuan dari proses ini adalah memperoleh parameter intrinsik kamera serta nilai distorsi lensa yang dibutuhkan dalam estimasi posisi dan orientasi objek terhadap kamera. Pada penelitian ini, metode Zhang digunakan sebagai pendekatan kalibrasi, yaitu dengan mengambil citra pola papan catur dari berbagai sudut pandang dan jarak. Papan catur yang digunakan memiliki konfigurasi sudut 9×6 dan dicetak di atas permukaan datar

agar tidak melengkung(12). Kamera Logitech C270 yang terpasang pada Raspberry Pi dihubungkan melalui *port* USB, lalu dijalankan kode kalibrasi menggunakan Python.

Selama proses akuisisi data, kamera diarahkan ke papan catur dari berbagai posisi dan sudut untuk mendapatkan variasi perspektif yang dibutuhkan dalam proses kalibrasi. Hasil dari proses kalibrasi meliputi matriks intrinsik kamera serta parameter distorsi, yang menjadi fondasi penting dalam sistem pengolahan citra berbasis estimasi pose. Parameter tersebut digunakan untuk mengoreksi distorsi dan meningkatkan akurasi dalam mendeteksi serta menentukan posisi ArUco marker secara *real-time* pada misi navigasi *visual* berbasis citra.

### 3.5 Uji Terbang *Object Detection*



Gambar 2. Diagram Alir Misi Terbang

Pada Gambar 2, merupakan susunan metode penyelesaian uji misi terbang. Gambar tersebut menggambarkan alur metodologi yang digunakan dalam pengembangan sistem pengolahan citra berbasis quadcopter untuk misi pendeteksian objek menggunakan ArUco marker. Tahapan dimulai dari aktivitas awal berupa studi literatur, yang bertujuan untuk memperoleh pemahaman mendalam mengenai konsep dan teknologi yang mendasari pengembangan sistem. Selanjutnya dilakukan penentuan kebutuhan sistem, yang mencakup pemilihan parameter dan spesifikasi teknis yang mendukung pencapaian misi.

Tahap berikutnya adalah perakitan komponen perangkat keras quadcopter, dilanjutkan dengan proses kalibrasi seluruh subsistem seperti akselerometer, kompas, radio, dan ESC. Setelah kalibrasi selesai, dilakukan uji terbang manual (perdana) untuk memastikan seluruh komponen berfungsi dengan baik. Sistem kemudian melalui proses penyetelan parameter kendali menggunakan fitur *PID Autotune*.

Setelah quadcopter menunjukkan performa terbang yang stabil, langkah selanjutnya adalah pengembangan sistem

pengolahan citra serta integrasi program *autonomous*. Sistem ini memungkinkan quadcopter untuk melakukan deteksi objek secara mandiri dan menjalankan navigasi berbasis *visual*. Pengujian misi dilakukan untuk mengevaluasi kemampuan quadcopter dalam mendeteksi ArUco marker dan melakukan pendaratan tepat di atasnya. Jika *marker* berhasil terdeteksi dan pendaratan berlangsung sesuai target, maka data dari keseluruhan proses pengujian dianalisis untuk menilai kinerja sistem secara menyeluruh.

## 4. HASIL DAN PEMBAHASAN

### 4.1 Hasil Perakitan Quadcopter

Hasil akhir dari proses perakitan menunjukkan bahwa seluruh komponen utama Quadcopter v10 telah terintegrasi dengan baik dalam konfigurasi rangka X. *flight controller* Pixhawk ditempatkan di bagian tengah rangka dengan peredam getaran (*dampner*) untuk meminimalkan gangguan dari vibrasi motor. Sistem propulsi terdiri dari empat motor *brushless* yang terhubung ke *Electronic Speed Controller* (ESC) dan *Power Distribution Board* (PDB), serta dikendalikan melalui komunikasi langsung dengan *flight controller*. Sistem tenaga, navigasi, dan komunikasi; termasuk baterai, GPS, dan modul telemetri telah terpasang secara fungsional dan saling terhubung. Hasil perakitan quadcopter terlampir pada Gambar 3.



Gambar 3. Hasil Perakitan Quadcopter

Dari hasil perakitan ini, diperoleh *platform* UAV yang telah siap untuk tahap kalibrasi, pengujian terbang, dan integrasi sistem pengolahan citra. Tata letak komponen yang rapi dan terorganisir menjadi fondasi penting untuk memastikan performa penerbangan yang stabil serta efisiensi dalam proses *debugging* dan perawatan.

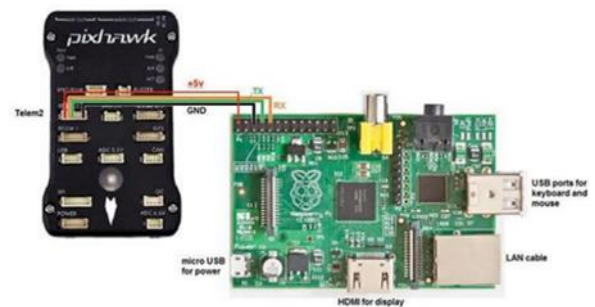
### 4.2 Hasil Kalibrasi Kamera

Perancangan koneksi antara Raspberry Pi dan laptop dilakukan untuk mendukung proses pemantauan serta pengendalian sistem pengolahan citra secara efisien(13). Raspberry Pi berfungsi sebagai unit pemrosesan utama yang menjalankan program deteksi *marker*, sedangkan laptop digunakan sebagai terminal akses jarak jauh tanpa

memerlukan perangkat *input-output* langsung pada papan. Akses dilakukan melalui jaringan Wi-Fi lokal menggunakan protokol komunikasi berbasis SSH (*Secure Shell*) dan aplikasi VNC *Viewer*, sehingga dapat dilakukan pengelolaan file, menjalankan skrip, serta melakukan proses *debugging* secara *real-time* dari laptop. Koneksi ini memungkinkan fleksibilitas pengoperasian, sekaligus mempermudah integrasi sistem saat pengujian di lapangan.

### 4.3 Hasil Perancangan Sistem *Autonomous*

Integrasi antara Raspberry Pi dan *flight controller* Pixhawk merupakan elemen penting dalam implementasi sistem navigasi otonom. Kedua perangkat dihubungkan melalui koneksi serial menggunakan *port* telemetri pada Pixhawk dan antarmuka GPIO (*General Purpose Input/Output*) pada Raspberry Pi. Jalur komunikasi terdiri dari tiga pin utama, yaitu RX, TX, dan GND yang dihubungkan secara silang untuk mendukung pertukaran data dua arah(14). Setelah koneksi fisik terpasang, pengujian dilakukan melalui terminal untuk memastikan komunikasi berhasil, salah satunya dengan mengirim perintah *arming* dari Raspberry Pi ke Pixhawk. Komunikasi antarperangkat ini dimediasi oleh protokol MAVLink, yang memungkinkan *transfer* perintah dan data navigasi secara efisien. Dengan integrasi ini, sistem pengolahan citra yang berjalan pada Raspberry Pi dapat memberikan instruksi langsung kepada *flight controller*. Gambar 4 menampilkan skematik rangkaian Raspberry Pi dengan Pixhawk.

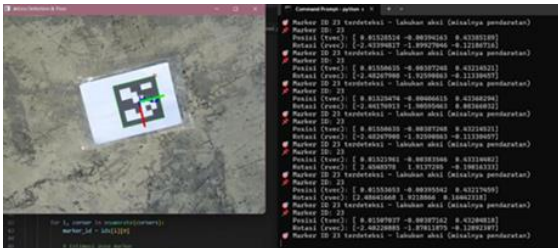


Gambar 4. Skematik Rangkaian Raspberry Pi dengan Pixhawk

### 4.4 Hasil Pemrograman Pengolahan Citra

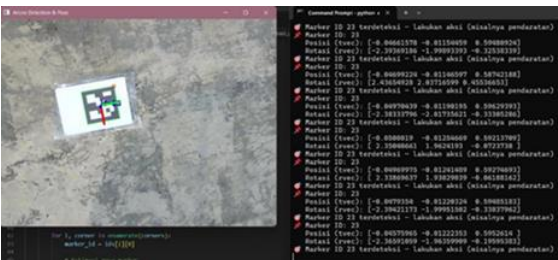
Pemrograman sistem pengolahan citra dilakukan menggunakan bahasa Python versi 3.11 dengan memanfaatkan pustaka OpenCV dan modul ArUco untuk keperluan deteksi *marker* serta estimasi pose(15). Proses ini melibatkan dua tahapan utama, yaitu identifikasi ID *marker* dan perhitungan estimasi posisi relatif *marker* terhadap kamera. *Marker* yang digunakan berasal dari kamus DICT\_6X6\_250, dengan ID yang ditargetkan adalah 23. Sistem dibangun agar dapat bekerja secara *real-time*, di mana citra yang ditangkap oleh kamera Logitech C270 diolah langsung oleh Raspberry Pi untuk mendeteksi keberadaan *marker* dan menghitung pose *marker* terhadap sistem koordinat kamera. Hasil implementasi menunjukkan bahwa sistem berhasil mendeteksi *marker* ID 23 secara konsisten pada tiga variasi ketinggian, yaitu 1.5 meter, 2 meter, dan 3 meter.

Pada pengujian sistem deteksi objek menggunakan ArUco marker ID 23 pada ketinggian 1.5 meter yang ditampilkan pada Gambar 5, sistem berhasil mengenali *marker* secara konsisten dan menghitung informasi posisi serta orientasi *marker* terhadap kamera dalam bentuk vektor translasi (*tvec*) dan rotasi (*rvec*). Salah satu data hasil pengujian menunjukkan nilai *tvec* sebesar [0.01528541, -0.09301766, 0.43838189], yang mengindikasikan bahwa *marker* terletak sekitar 1,5 cm di sebelah kanan kamera, 9,3 cm di bawah.



Gambar 5. Deteksi *Marker* Ketinggian 1.5 meter

Pada Gambar 6, dapat dilihat hasil pengujian ketinggian 2 meter, sistem kembali menunjukkan kemampuan dalam mendeteksi ArUco marker ID 23 secara akurat dan *real-time*. Salah satu hasil estimasi pose yang terekam memperlihatkan nilai vektor translasi (*tvec*) sebesar [0.04669922, -0.01126489, 0.61134475], yang berarti *marker* terdeteksi berada sekitar 4,6 cm di sebelah kanan kamera, 1,1 cm di bawah. Ini menunjukkan bahwa *marker* masih berada dalam jangkauan pandang kamera dan cukup ideal untuk dilakukan proses navigasi atau pendaratan. Sementara itu, vektor rotasi (*rvec*) yang dihasilkan adalah [-2.39421713, -1.99915802, -0.3095072], yang menggambarkan orientasi *marker* terhadap kamera dalam bentuk rotasi tiga dimensi.



Gambar 6. Deteksi *Marker* Ketinggian 2 meter

Dapat dilihat pada Gambar 7, dimana pada ketinggian 3 meter, sistem pengolahan citra tetap menunjukkan hasil pendeteksian ArUco marker ID 23. Salah satu hasil deteksi mencatat nilai vektor translasi (*tvec*) sebesar [-0.08068189, -0.01698085, 0.68316147]. Artinya, *marker* terdeteksi berada sekitar 8 cm di sebelah kiri kamera, 1,7 cm lebih rendah. Meskipun ketinggian meningkat, sistem masih mampu mengenali posisi *marker* secara akurat dalam ruang tiga dimensi. Vektor rotasi (*rvec*) yang menyertainya adalah [-2.23477152, -2.08706678, 0.12381246], yang menggambarkan orientasi *marker* terhadap kamera dalam bentuk rotasi.

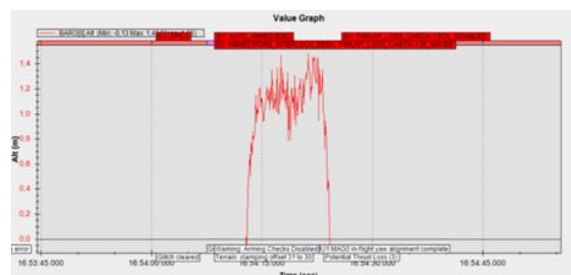


Gambar 7. Deteksi *Marker* Ketinggian 3 meter

#### 4.5 Hasil Uji Misi Terbang *Object Detection*

Uji misi terbang *object detection* dilakukan untuk menguji kemampuan sistem pengolahan citra pada Quadcopter v10 dalam mendeteksi ArUco marker dan melakukan pendaratan secara otomatis. Pengujian ini dilakukan di ruang terbuka, menggunakan ArUco marker berukuran 15 cm × 15 cm yang dicetak pada media *art paper* ukuran A4, kemudian diletakkan rata di permukaan sebagai target pendaratan. Sebelum uji terbang dilakukan, sistem pemrograman pengolahan citra telah dirancang menggunakan bahasa Python 3.11 yang mengintegrasikan pustaka OpenCV dan ArUco.

Program ini memungkinkan quadcopter untuk mendeteksi *marker* berdasarkan ID, menghitung estimasi posisi (*pose estimation*) menggunakan fungsi `cv2.aruco.estimatePoseSingleMarkers()`, dan memberikan instruksi pendaratan berdasarkan hasil deteksi *visual* secara *real-time*. Proses pengujian dimulai dengan quadcopter melakukan *take off* secara otomatis dan memasuki fase *hover* pada ketinggian yang telah ditentukan. Selama melayang, kamera yang terpasang di bagian bawah quadcopter akan menangkap citra dan mengirimkannya ke Raspberry Pi untuk diproses. Saat *marker* terdeteksi, sistem mencatat data posisi (*tvec*) dan orientasi (*rvec*) dari *marker* terhadap kamera, serta memicu aksi pendaratan tepat di atas *marker*. Pada Gambar 8, terlampir data *altitude* uji terbang *object detection* pada ketinggian 1.5 m.

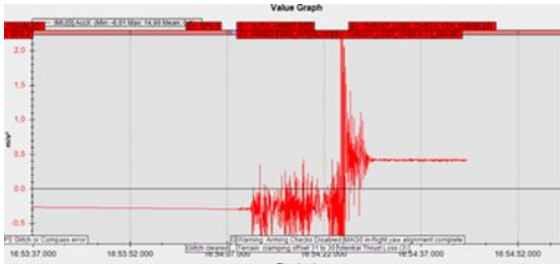


Gambar 8. Grafik *Altitude vs Time*

Pada awal grafik, terlihat bahwa quadcopter berada dalam kondisi diam dengan nilai ketinggian mendekati nol, setelah perintah *auto take off* dijalankan. Selama fase *hovering*, grafik menunjukkan fluktuasi ketinggian yang masih dalam batas toleransi sistem, menandakan bahwa quadcopter mampu mempertahankan posisi terbang. Pada saat inilah sistem pengolahan citra mulai bekerja untuk mendeteksi ArUco marker yang berada di bawah. Setelah *marker* berhasil dikenali, sistem kemudian mengeksekusi perintah

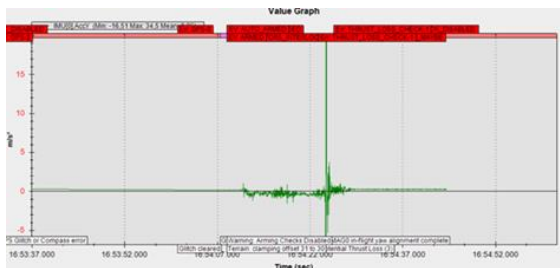
pendaratan otomatis, yang ditunjukkan oleh penurunan ketinggian secara bertahap hingga kembali ke permukaan.

Gambar 9 memperlihatkan grafik percepatan pada sumbu X terhadap waktu selama uji misi *object detection*. Hasil analisis grafik menunjukkan bahwa sistem quadcopter berhasil mendeteksi objek secara *real-time* dan memberikan respons yang cepat melalui perubahan signifikan pada kecepatan longitudinal. Fluktuasi tajam pada  $V_x$  serta aktivasi sistem *failsafe* menunjukkan bahwa sistem navigasi dan deteksi bekerja secara responsif dalam menghadapi potensi bahaya di jalur terbang.



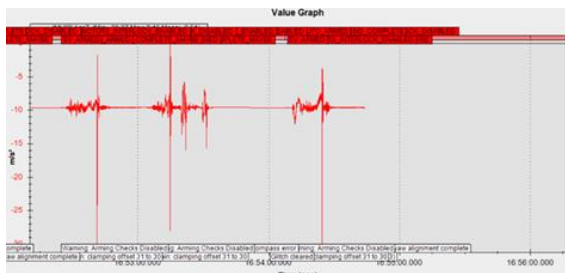
Gambar 9. Grafik  $V_x$  vs Time

Gambar 10 menampilkan grafik percepatan sumbu Y terhadap waktu selama uji misi *object detection* pada ketinggian 1.5 meter. Puncak lonjakan percepatan yang cukup tajam sesaat terjadi, hal ini kemungkinan dipicu oleh manuver mendadak seperti koreksi arah atau gangguan angin.



Gambar 10. Grafik  $V_y$  vs Time

Gambar 11 menunjukkan grafik percepatan sumbu Z terhadap waktu pada saat uji terbang di ketinggian 1.5 meter. Nilai percepatan tampak bernilai negatif, yang merupakan hasil dari konvensi sistem koordinat IMU, di mana sumbu Z positif mengarah ke bawah.



Gambar 11. Grafik  $V_z$  vs Time

Saat quadcopter berada dalam kondisi diam, nilai mendekati  $-9.8 \text{ m/s}^2$  akibat gaya gravitasi. Fluktuasi lebih besar terjadi

saat fase *take off*, di mana percepatan ke atas menghasilkan nilai negatif.

## 5. KESIMPULAN

Quadcopter v10 dengan konfigurasi rangka X dan sistem *flight controller* Pixhawk 2.4.8, yang mampu melakukan penerbangan secara *autonomous* untuk misi *object detection*. Sistem dilengkapi dengan Raspberry Pi 4 sebagai pemroses utama pengolahan citra dan kamera Logitech C270 sebagai input *visual* berhasil dirakit.

Sistem pengolahan citra berbasis ArUco marker berhasil dirancang dan diimplementasikan menggunakan bahasa pemrograman Python 3.11 dan pustaka OpenCV. Sistem ini mampu mendeteksi marker ID 23 serta melakukan estimasi pose *marker* dalam bentuk vektor rotasi dan translasi secara *real-time*.

Uji terbang untuk misi *object detection* dapat dilakukan, pada ketinggian 1.5 m dan 2 m quadcopter dapat mendarat tepat di atas *marker*. Namun pada ketinggian 3 m, quadcopter belum bisa mendarat di atas *marker*. Hal ini kemungkinan dipengaruhi oleh faktor lingkungan sekitar seperti angin yang cukup kencang.

## DAFTAR PUSTAKA

1. Lebedev I, Erashov A, Shabanova A. Accurate Autonomous UAV Landing Using Vision-Based Detection of ArUco-Marker. *Lect Notes Comput Sci (including Subser Lect Notes Artif Intell Lect Notes Bioinformatics)*. 2020;12336 LNAI:179–88.
2. Utami D. Quadcopter V9 : Kaji Pengolahan Citra Untuk Misi Terbang Object Detection. Politeknik Negeri Bandung. 2024.
3. Hartono B, Rizki Zuhri M, Asti Rosalia C, Fauzan N. Autonomous Quadcopter Image Processing for Simulated Search and Rescue Flights. *Met J Sist Mek dan Termal [Internet]*. 2023;7(2):8–17. Available from: <http://metal.ft.unand.ac.id/index.php/metal/article/view/263>
4. oxygen. OpenCV. Detection of ArUco Markers. Available from: [https://docs.opencv.org/4.x/d5/dae/tutorial\\_aruco\\_detection.html](https://docs.opencv.org/4.x/d5/dae/tutorial_aruco_detection.html)
5. Hakim T, Priambodo AS. Navigasi Quadcopter Berbasis ArUco Marker dengan OpenCV. *J List Instrumentasi, dan Elektron Terap*. 2025;6(1):28.
6. Muhamad A, Panjaitan SD, Yacoub RR. Design and Development of Flight Controller for Quadcopter Drone Control. *Telecommun Comput Electr Eng J*. 2024;1(3):279.
7. Dijaya R. *Buku Ajar Pengolahan Citra Digital*. Sidoarjo: UMSIDA Press; 2023. ISBN: 978-623-464-075-5.
8. Zhong X, Liang Y. Raspberry Pi : An Effective Vehicle in Teaching the Internet of Things in Computer Science and Engineering. 2016;
9. Pătru GC, Pîrvan AI, Rosner D, Rughiniș RV. Fiducial Marker Systems Overview and Empirical Analysis of Aruco, Apriltag and Cctag. *UPB Sci Bull Ser C Electr Eng Comput Sci*. 2023;85(2):49–62.
10. Dwi. IDMETAFORA. 2022. Mengenal Prinsip, Jenis, Bagian, dan Komponen Drone. Available from: <https://idmetafora.com/news/read/997/Mengenal-Prinsip->

- Jenis-Bagian-dan-Komponen-Drone.html
11. Siki Z, Takács B. Automatic Recognition of ArUco Codes in Land Surveying Tasks. 2021;9(1):115–25.
  12. OpenCV: Camera Calibration [Internet]. [cited 2025 Jul 7]. Available from: [https://docs.opencv.org/3.4/dc/dbb/tutorial\\_py\\_calibration.html](https://docs.opencv.org/3.4/dc/dbb/tutorial_py_calibration.html)
  13. Mathe SE, Kondaveeti HK, Vappangi S, Vanambathina SD, Kumaravelu NK. A comprehensive review on applications of Raspberry Pi. *Comput Sci Rev* [Internet]. 2024;52(October):100636. Available from: <https://doi.org/10.1016/j.cosrev.2024.100636>
  14. Raspberry Pi Companion with Pixhawk | PX4 Guide (main) [Internet]. [cited 2025 Jul 7]. Available from: [https://docs.px4.io/main/en/companion\\_computer/pixhawk\\_rpi.html](https://docs.px4.io/main/en/companion_computer/pixhawk_rpi.html)
  15. Srinath KR, Associate. Python – The Fastest Growing Programming Language K. *J Sci Med Sport*. 2017;20(7):678–83.