Vision-Based Autonomous UGV Detection, Tracking, and Following for a UAV

Conference Paper · January 2024

DOI: 10.2514/6.2024-2093

5 authors, including:

Fatma Amil
Cranfield University
2 PUBLICATIONS **1** CITATION
SEE PROFILE

Hüseyin burak Kurt
TAI - Turkish Aerospace Industries, Inc.
8 PUBLICATIONS **26** CITATIONS
SEE PROFILE

Semih Beyçimen
Bursa Uludağ Üniversitesi
8 PUBLICATIONS **29** CITATIONS
SEE PROFILE

Murat Millidere
Cranfield University
21 PUBLICATIONS **73** CITATIONS
SEE PROFILE

Vision-Based Autonomous UGV Detection, Tracking, and Following for a UAV

Fatma Gul Amil *, Muhammet Sen †, Huseyin Burak Kurt, ‡, Semih Beycimen § and Murat Millidere ¶
Cranfield University, College Rd, Wharley End, Bedford MK43 0AL
Turkish Aerospace, Ankara, Turkiye

This study proposes a methodology for unmanned ground vehicle (UGV) navigation in off-road environments where GPS signals are not available. The Husky-A200 at Cranfield University, United Kingdom has been used as a UGV in this research project. Due to the limited field of vision of UGVs, a UAV-UGV collaboration approach was adopted. The methodology involves five steps. The first step is divided into three phases: The aerial images of UGV from UAV are generated in the first phase. In the second phase, the UGV is detected and tracked using computer vision techniques. In the third phase, the relative pose (position and heading) between the UAV and UGV is estimated continuously using visual data. In the second step, the UAV maintain a fixed location (position and heading) relative to the UGV. The third step involves capturing aerial images from the UAV's mounted camera and transmitting it to the ground station instantly to create a global traversability map that classifies terrain features based on their traversability. In the fourth step, additional sensors such as LiDAR, radar, and IMU are used to refine the global traversability map. In the final step, the UGV navigates automatically using the refined traversability map. This study will focus on the first two steps of the methodology, while subsequent studies will address the remaining steps.

Nomencleature

UGV	=	Unmanned Ground Vehicle
UAV	=	Unmmaned Air Vehicle
C_p	=	pressure coefficient
Cx	=	force coefficient in the x direction
dt	=	time step

I. Introduction

The concept of UAV-UGV cooperative systems, where aerial vehicles provide a wide-range view to assist ground vehicles' navigation and operation, has been a hot research topic in recent years. The UAV can provide valuable perspective and broad visual coverage, while the UGV performs tasks on the ground. Unmanned aerial vehicles (UAVs) and unmanned ground vehicles (UGVs) have been widely used in various fields, including target localization [1], formation control of UGVs [2], autonomous following and landing on a moving target [3] [4] [5] [6] [7] [8], construction [9], and search and rescue [10] [11], due to their ability to perform tasks in hazardous or inaccessible environments. However, navigating UGVs in GPS-denied off-road environments is still a challenging problem. In this paper, we propose a new methodology that combines the use of a UAV and a UGV for autonomous off-road navigation. The UAV acts as an external eye for the UGV, providing enhanced situational awareness via its higher vantage point. The UGV is detected, and tracked using computer vision algorithms and the relative pose between the UAV and the UGV is estimated. The UAV follows the UGV autonomously at a predefined distance and altitude. And then aerial images are sent to the UGV for generating global traversability. Additionally, the UGV will use its sensors, including Lidar, Radar,

*PhD Candidate, Centre for Aeronautics
†PhD Candidate, Centre for Autonomous and Cyberphysical Systems
‡Flight Dynamics, Simulation, and Control Engineer
§Research Fellow, Centre for Autonomous and Cyberphysical Systems
¶Resarch Fellow, Centre for Aeronautics, College Rd, Wharley End, Bedford MK43 0AL

and IMU, to tune the global traversability generated by the processor. In the final step, the refined map is used by the UGV to navigate automatically in the off-road environment. In this paper, the first two steps will be covered.

The first step is divided into three phases: The main objective of the first phase is to develop a dataset of UGV using aerial images from UAVs in different rural environments. In the second phase, the dataset is used to train machine learning models for UGV detection and tracking in off-road environments. In the third phase, the relative pose (position in x-y-z direction and heading angle) is estimated.

The first key to the success of UAV-UGV cooperation is the development of robust object detection and tracking algorithms. Computer vision techniques to detect the moving target using onboard cameras was widely studied in literature [12]. Some examples are the pad detection algorithms in Ref. [12]and the vehicle detection algorithms in Ref. [13] and the marine vehicle detection algorithms in Ref. [14]. There are two types of detection algorithms: single-stage detectors and two-stage detectors [15]. The single-stage detectors that process object identification and classification in a single pass through the network whereas two-stage detectors first identify areas of interest and then classify them. Single-stage detectors offer faster and more efficient detection capability compared to two-stage object detection algorithms, such as Faster RCNN[16]. YOLO[17] is one type of single-stage detectors and in this study, YOLO is chosen as a detection and tracking algorithm due to reduced computational time requirement.

Object tracking is another important task in real-time object detection, where the goal is to estimate the motion of the target object over time. ByteTrack [18] is a popular approach for object tracking that is simple yet effective. It utilizes YOLOX and BYTE, high-performance object detectors, for data association. BYTE takes into account all detection results, regardless of their detection scores, to improve the performance of ByteTrack. By taking into account every detection box, including objects with lower detection scores, it can recover objects that might otherwise be missed [18]. ByteTrack demonstrates robustness in dealing with challenges such as occlusion, motion blur, and variations in object size, which results in accurate object tracking.

In this process, the next task is to feed the spatial data, derived from the detection and tracking of the UGV using YOLO and ByteTrack, into algorithms specifically designed for relative pose estimation. These algorithms are tasked with calculating the position and orientation of the UGV, with the perspective anchored on the UAV. This calculation is intrinsically dependent on the spatial data obtained from tracking the UGV.

Over the years, the techniques involved in pose estimation have seen significant advancements. Starting from geometry-based approaches, such as the eight-point algorithm [19], the field has evolved to incorporate state-of-the-art methods that leverage machine learning, more specifically, deep learning-based pose estimation [20]. An illustrative example of this evolution is found in Ref[21], where specific markers were employed for ground vehicle pose estimation. This estimation process is sophisticated because it transforms 2-dimensional image data from the UAV's cameras into a 3-dimensional understanding of the UGV's pose in the world. This transformation involves interpreting the depth or distance of the UGV from the UAV in addition to its orientation, which is challenging due to the inherent complexity of extracting 3D information from 2D images. Thus, the role of relative pose estimation in this context is crucial to accurately navigate and coordinate the actions of the UGV and UAV in their shared environment.

The second step of the methodology involves the UAV following the UGV at a predefined position and heading. The UAV serves as a tracker for the UGV, which is useful in scenarios where the UGV needs to navigate through complex environments with obstacles that may obstruct its view. By following the UGV, the UAV can provide a bird's eye view of the terrain and help the UGV navigate through obstacles. The predefined distance and altitude at which the UAV follows the UGV can vary based on the environment and the capabilities of the UAV and UGV. In general, the UAV should follow the UGV at a distance that provides a clear view of the terrain while avoiding collisions with obstacles. Similarly, the altitude at which the UAV flies should be high enough to provide a wide view of the terrain but not so high that the details of the terrain are lost. The most important requirements for the autonomous flight of a quadrotor are effective position control, altitude control, and attitude control (stabilization) [22]. However, quadrotors are highly complex and difficult-to-control vehicles due to their non-linear, underactuated, multivariable and unstable nature [23].

The literature review [24] indicates that most common quadrotor control approaches are mostly classified into two as linear, and nonlinear. Linear control strategies are based on linear models of system dynamics around the desired operating point. Linear control methods include Proportional–Integral– Derivative (PID) [25] [26], Linear Quadratic Regulator (LQR) [8][27] [28] [29][30] and Proportional-Derivative (PD) [31] [32] [33]. The most common nonlinear methods include Backstepping (BS) [34] , Sliding Mode Control (SMC) [8][35][30], nonlinear H-infinity (H) [36], and inverse dynamic control [37].

In this study, control algorithms have been run and tested on embedded electronic systems. These systems are often called flight controllers. Along with attitude, position, and altitude control, they also perform tasks such as reading sensor values and communicating with the ground station. Embedded systems used in such applications can be

classified as FPGA-based, arm-based, Atmel-based, and raspberry pi-based [24]. Pixhawk PX4 and Pixhawk 2 are ARM-based.The Pixhawk and its autopilot software Px4 have been selected for this project mainly because they are both widely adopted by academic, and developer communities for their flexibility, low cost and reliable autopilot system. The Px4 uses P-PID controllers for its stability augmentation system, attitude and position controller. Even though Px4 allows one to change the flight controller structure, the proportional-integral-derivative (PID) format is still by far the most popular choice [38]. The PID method calculates the error between the controlled signal and the desired reference value, then applies a correction based on the error's P, I, and D components. The PID approach has certain advantages that cannot be overlooked [32] [39]: 1) easy implementation, 2) flexible parameter adjustment, 3) easy design, and 4) satisfactory performance.

In summary, the use of UAVs and aerial images for UGV detection, tracking, and navigation has become increasingly popular in recent years. This study aims to contribute to the existing literature by developing a comprehensive dataset for UGV detection and tracking using aerial images from UAVs in different rural environments. The dataset will be used to train machine learning models for UGV detection and tracking in off-road environments where GPS is not available. This paper describes the proposed methodology in detail and discusses its potential applications and benefits. The remainder of the paper is organized as follows: Section II provides an overview of related work. Section 3 describes the proposed methodology in detail. Section 4 presents the experimental results and discusses the performance of the proposed method. Finally, Section 5 concludes the paper and discusses future work.

II. Off-Road UGV Dataset in Rural Environments

The Off-Road Unmanned Ground Vehicle (UGV) Dataset in Rural Environments marks a significant contribution to the research studies. It is important to note that data collection forms the backbone of this effort. This process involves the use of a specialized platform, the Husky-A200 from Clearpath Robotics, which employs unique sensors to gather real-time data. For safety reasons, this platform is manually controlled via a joystick throughout the data collection process.

As soon as the drone is switched ON, it undergoes its factory-defined preflight checks such as the GPS reception (as it's positioning is entirely dependent on the GPS), efficient working of inbuilt IMU, flight control system, motors, camera system and gimbal. The drone is connected via 2.4GHZ frequency with the transmitter. In order to view the camera feed from the drone, the transmitter can either be connected with a smart phone / tablet or iPad. The drone compass is calibrated prior to the flight to reduce any compass invariances. Once all the checks have been satisfied the drone is all set to take off.

The drone takes off and loiters at an initial height of 5m above the Husky. The altitude is gradually increased to 10m so that the drone can capture the Husky. The drone is always aligned in such a way that the Husky is always focused at the centre of the frame. Once this parameter is satisfied the camera recording is toggled and as the Husky moves, the drone maneuvers parallelly above it at relatively same speed. Then the height is gradually increased to 12m, 15m, 18m and 20m respectively with the same condition of focusing Husky at the centre of the frame and moving parallelly with relatively same speed.

III. Methodology

In this work, we address two fundamental stages. The first stage focuses on the real-time detection and tracking of the ground vehicle, which poses significant challenges due to the variable speeds of the UGV and the complexity of environmental conditions. This involves differentiating the UGV from similar objects in its vicinity, a task that demands both safety and simultaneity in approach.

The focus of the second stage was on pose estimation and control, which are critical for maintaining and optimizing the UAV-UGV interaction. This stage involves the process of estimating the relative pose of the UGV using sophisticated visual-based algorithms. Accurate pose estimation is essential for effective UAV control, enabling the aerial vehicle to adapt to the UGV's movements and to navigate challenging terrains accurately. This step is crucial for enhancing the autonomy and coordination of the UAV-UGV system, especially in environments where conventional GPS and attitude control mechanisms are ineffective or unavailable. The detailed methodology for these stages, including the specific techniques and technologies employed, is outlined below.

1) **Data Collection and Preprocessing**: Aerial datasets were collected in various off-road conditions on which UAV and UGV platforms are operated manually. Following the collection phase, a detailed data labeling process was

conducted using MATLAB image labeling and the Roboflow labeling toolbox, which was critical to accurately training our models for UGV detection.

2) **UGV Detection using Deep Learning**: Implementing the YOLOv7 algorithm to detect Husky UGV by training the annotated dataset. For real-time implementation, ROS-wrapper was used with the best pre-trained model which was trained several times using different parameters and data sizes.
3) **UGV Tracking**: Utilizing ByteTrack for continuous tracking of the UGV and ensuring robust tracking under varying environmental conditions and UGV speeds.
4) **Relative Pose Estimation**: Deploying visual-based algorithms to estimate the relative position and orientation between the UAV and UGV using Aruco markers and integrating data from UAV's onboard sensors to enhance pose estimation accuracy.
5) **UAV-UGV Collaborative Operation**: Establishing a communication protocol for UAV and UGV coordination and developing real-time algorithms for the UAV to maintain at desired relative position to the UGV, adapting to its movements and the terrain.
6) **Position and Heading Control of UAV**: Ensuring precise UAV orientation and position relative to a UGV with pose estimation and geometric data using a PID-based controller.
7) **Testing and Validation**: Conducting field tests in various off-road environments to assess system robustness and reliability. Evaluating the system's performance in terms of detection accuracy, tracking stability, and navigation efficiency.

The proposed system relies on advanced computer vision algorithms and the capacity for real-time image processing. The absence of GPS presents a challenge in achieving precise monitoring; however, through carefully considered design and rigorous testing, reliable performance can still be attained. This approach underscores the potential for sophisticated technology to effectively compensate for limitations in conventional navigation systems.

A. System Description

In this study, Husky A200 and Parrot AR. Drone 2.0 were used for data collection, validation and testing and a mini ITX computer with Nvidia GTX 1065 GPU in the platform was utilised for all progressing steps. The platform was driven by a Logitech F710 joystick for manual progress such as data collection. Fig. 3 illustrates the Husky A200 snapshot with Velodyne VLP 16 Lidar, 3 Stereolab Zed-2 Stereo Camera, Duro RTK GPS, 24V 20Ah Sealed Lead Acid battery and Fig. 1 shows our methodological architecture.

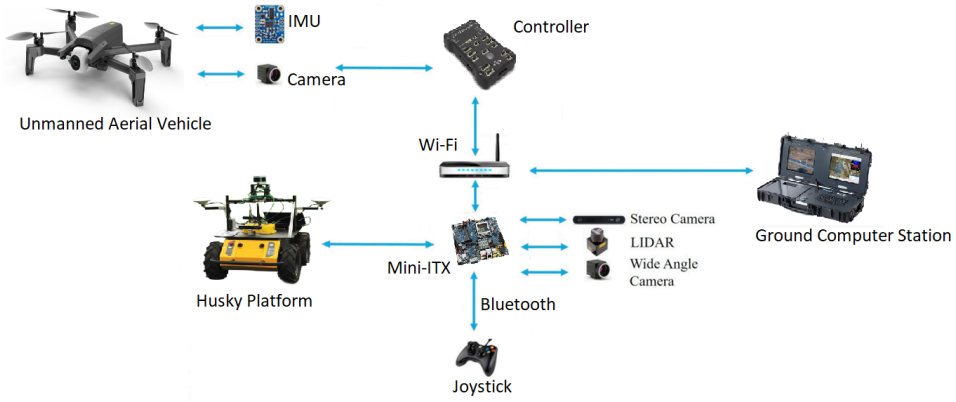

Fig. 1 System Overview

The UAV used is Parrot Anafi work extended model and operated by sky controller 3 with an operational frequency range of 2.4GHz - 5.8GHz. The vision system includes the inbuilt 21MP camera with a maximum 3X digital zoom feature. The drone streams its live video feed via WiFi to the Ground Control Station (preferably smartphone/iPad/Tablet).

Fig. 2 Unmanned Air Vehicle, Parrot Anafi

Fig. 3 Husky A200 Vehicle Platform snapshot with VLP 16 Lidar, Stereo Camera, Duro RTK GPS sensors payload (Centre for Autonomous & Cyber-Physical Systems, Ground autonomy Lab, Cranfield University)

B. UGV Detection, Tracking and Visual-Based Relative Pose Estimation

This section addresses the field of computer vision, with a focus on detecting and tracking objects for unmanned ground vehicles using aerial images. A critical aspect of this process is object detection, which involves identifying objects and determining their locations within the image. The detection algorithm's primary task is to estimate the boundaries around objects and assign relevant class labels to them. This step is crucial for successful tracking. Object detection plays an important role in the overall algorithm, as many tracking algorithms rely on the tracking-by-detection approach.

1. Object Detection

In this study, the deep learning algorithm to be used for the object detection algorithm is YOLOv7. It is based on the YOLO (You Look Only Once) framework, a single-stage detector that processes object identification and classification in a single pass through the network. This offers a faster and more efficient detection method compared to two-stage object detection algorithms, such as Faster RCNN [16], which first identify areas of interest and then classify them. Due to its reduced computational time requirements, YOLO is better suited for real-time applications.

YOLO has been developed since 2016 with various variations [40] [41]. In July 2022, Chien-Yao Wang, Alexey Bochkovskiy, and Hong-Yuan Mark Liao published the official YOLOv7 article titled "YOLOv7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors" [17]. In this research, YOLOv7 is chosen for its advantages which have been mentioned in the literature[42].

The YOLOv7 network, used in tasks like identifying and locating objects in images, has a special feature called an Extended Efficient Layer Aggregation Network, or E-ELAN for short. This is essentially the core structure, or 'backbone,' that gives the system its strength. Basically, E-ELAN ensures that vital information is delivered faster and more efficiently, thereby improving the overall performance of YOLOv7 in its object detection tasks.

Evaluation Metrics The performance of object detection models is often evaluated using several criteria including precision, recall, mean Average Precision (mAP), Intersection Over Union (IoU), and the F1 score. These metrics are computed based on the ratios of true positives, false positives, and false negatives predicted by the model, as well as the geometric overlap between predicted and actual bounding boxes (for IoU), thus providing a multifaceted assessment of the model's performance.

Evaluation metrics are calculated as:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{1}$$

$$Precision = \frac{TP}{TP + FP} \tag{2}$$

$$Recall = \frac{TP}{TP + FN} \tag{3}$$

where TP means True-Positive, TN means True-Negative,FP means False-Positive,FN means False-Negative.

$$F1 = \frac{2 * Precision * Recall}{Precision + Recall} = \frac{2 * TP}{2 * TP + FP + FN} \tag{4}$$

Average Precision is calculated as:

$$\text{AP} = \sum_{n} (R_n - R_{n-1}) P_n \tag{5}$$

where R_n and P_n are the precision and recall at the nth threshold. AP can also be regarded as the area under the precision-recall curve. MAP is the mean of AP over all the queries.

In the Table 1 shows the state of art object detection algorithms which are compared with the efficiency using their evaluation metrics. YOLOv7 demonstrates significant improvements in efficiency compared to its previous version, YOLOv5[42]. It requires 63% less computational processing power while achieving a 47% faster inference speed. This indicates that YOLOv7 can deliver higher performance in object detection tasks while consuming fewer resources and providing faster results than YOLOv5. Remarkably, YOLOv7 achieves a 2% higher accuracy and a 509% faster inference rate compared to the best-performing Cascade-Mask R-CNN [43] models, which have previously achieved significantly higher detection accuracy using multi-stage architectures. Due to its state-of-the-art performance, the YOLOv7 object detection algorithm is utilized in this study.

Table 1 The state of art object detection algorithms and their comparisons

Algorithm	Backbone	mAP (%)	Inference Time (ms)	FPS	Memory (MB)	Input Resolution	Detection Speed (Higher is better)	Dataset	Training Time (hours)
YOLOv3	Darknet-53	57.9	30	33	262	416x416	100	COCO	8
YOLOv4	CSPDarknet53	65.7	21	47	315	608x608	200	COCO	12
YOLOv5	CSPNet	68.7	20	50	300	640x640	230	COCO	10
YOLOv7	EfficientNet	72.0	16	62	330	640x640	270	COCO	14
SSD	VGG-16	48.5	35	28	180	300x300	80	COCO	100
Faster R-CNN	ResNet-50	58.0	90	11	350	800x800	60	COCO	16

2. Tracking

Object tracking is a fundamental task in computer vision that involves locating and following objects of interest in video sequences or image streams. The second step in the methodology is to track the detected unmanned ground vehicles across the sequence of images and estimate their position, velocity, and trajectory.

ByteTrack [44] will be used as a tracking algorithm in this study. It is a significant object-tracking algorithm, combining a high-performance detector with an advanced tracking-by-detection framework. Its unique feature includes an adaptive feature template that evolves over time, offering robustness against changes in object appearance. It uses a deep learning-based appearance model to manage complex scenes, enhancing performance in situations with occlusion and abrupt motion changes. ByteTrack also utilizes a Kalman Filter for predicting an object's future location, improving accuracy in tracking tasks. In addition to these features, ByteTrack's methodology stands out for its simplicity and general applicability in data association tasks. It diverges from traditional methods that conserve every detection box. Instead, ByteTrack retains most detection boxes, dividing them into high and low score categories. Initially, it links

short track segments (tracklets) to high-score detection boxes. However, due to factors like motion blur, occlusion, or size variation, some tracklets may be incorrectly matched. To rectify this, ByteTrack then associates these mismatched tracklets with lower-score detection boxes, effectively filtering out irrelevant background noise and enhancing object isolation.

Furthermore, ByteTrack's one-shot detection-based approach, which combine object tracking and detection in a single integrated model, is important for its efficiency. This integration allows for the sharing of computational resources between detection and tracking phases, thereby achieving higher tracking speeds without compromising on accuracy. This makes ByteTrack particularly well-suited for real-time applications where both speed and precision are of the essence.

3. Visual-Based Relative Pose Estimation

In the final phase of this methodology, we focus on estimating the relative pose through visual detection and tracking, initiated by the calibration of the UAV's onboard camera using MATLAB's calibration toolbox [45]. Calibration is crucial for accurately determining the camera's intrinsic parameters, such as focal length, optical center, and lens distortion, as well as its extrinsic parameters, including the rotation matrix *[R]* and translation vector *[t]* [46]. These parameters are crucial for defining the camera's orientation and position within the world coordinate system, which is fundamental for precise pose estimation. For calibration, a 9x6 checkerboard with 25mm square patterns is used. The camera's intrinsic parameters are formulated as a matrix, denoted as K,equation 6, and the extrinsic parameters are captured in the [R|t] matrix. Equation 7represents the camera's intrinsic parameters and the projection from 3D camera coordinates to 2D image coordinates.

$$\begin{bmatrix} fx & s & cx \\ 0 & fy & cy \\ 0 & 0 & 1 \end{bmatrix} = K \tag{6}$$

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} f_x & s & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} X_c \\ Y_c \\ Z_c \\ 1 \end{bmatrix} \tag{7}$$

The terms c_x and c_y represent the coordinates of the camera's optical center in pixel terms, located within an image, signifying the point where the optical axis intersects the image plane. The terms f_x and f_y indicate the camera's focal lengths along the x and y axes, respectively, determining the image's scale and affecting the perspective. The skew coefficient s is non-zero only when the image axes are not perpendicular, potentially leading to image distortion. The output u and v are the coordinates on the 2D image plane. This equation is crucial in camera calibration and in transforming 3D points to their corresponding 2D image coordinates. Furthermore, the world-to-camera coordinate transformation, outlined in Equation 8 , converts coordinates from the world frame (X_w, Y_w, Z_w) to the camera frame (X_c, Y_c, Z_c). This transformation is crucial for how a point in the real world is viewed from the camera's perspective.

$$\begin{bmatrix} X_c \\ Y_c \\ Z_c \end{bmatrix} = \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_x \\ r_{21} & r_{22} & r_{23} & t_y \\ r_{31} & r_{32} & r_{33} & t_z \end{bmatrix} \cdot \begin{bmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{bmatrix} \tag{8}$$

Where: (X_w, Y_w, Z_w) is the world frame, (X_c, Y_c, Z_c) is the camera frame and, *[R|t]* is the camera extrinsic matrix. R is a 3x3 rotation matrix, and t is a 3x1 translation vector.This matrix represents the camera's pose (position and orientation) in the world.

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} \sim \begin{bmatrix} f_x & s & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_1 \\ r_{21} & r_{22} & r_{23} & t_2 \\ r_{31} & r_{32} & r_{33} & t_3 \end{bmatrix} \begin{bmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{bmatrix} \tag{9}$$

After calibration, real-world 3D points are transformed into their corresponding 2D image projections by the camera, utilizing a scaling factor 's', along with matrices K and *[R|t]* as represented in equation 9. This calibrated camera

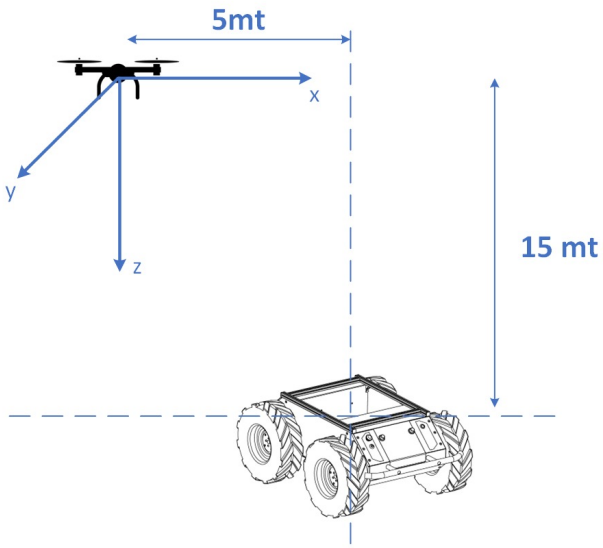

Fig. 4 The UGV following of the the UAV

setup is then used to detect specific patterns, such as Aruco markers, square patterns used for easy and accurate pose estimation on the UGV[47]. Marker detection involves converting video frames to grayscale and resizing for efficient processing in OpenCV, followed by pose estimation with cv2.aruco.estimatePoseSingleMarkers. This relationship is key to computing the relative pose of the UGV, using the intrinsic and extrinsic parameters to calculate the $[R|t]$ matrix representing the ground vehicle's pose -relative to the UAV's camera. These vectors are critical in determining the drone's pose in three dimensions. Rotation vectors are converted into Euler angles (yaw, pitch, and roll), offering a more intuitive representation of orientation. The translation vectors provide the position in a Cartesian coordinate system, facilitating the calculation of the drone's distance from the marker. Continuously monitoring the UGV's position, our system dynamically adjusts the UAV's flight path. This is facilitated by our DroneControl process, which feeds the pose information into the UAV's control system to guide its movements. The process enables dynamic adjustments based on the continuously updated pose data, ensuring precise control of the UAV in response to the UGV's movements.

C. Position and Heading Control of Quadcopter

The extraction of the yaw, pitch, and roll angles of the Husky vehicle through the drone's camera feed initiates the sequence of operations in our drone control system. Simultaneously, we maintain real-time updates on the distance between the Husky and the drone through continuous pose estimation. Initially, our drone control establishes a circular trajectory request from the Husky to the drone, guided by the ground vehicle's yaw angle. Fig.**??** illustrates the comparison between the Husky's rotation angles in the CW direction, ranging from 0 to 60 degrees, and the values obtained via pose estimation. Utilizing these findings, we derived a sixth-degree relative multiplier, employing this equation to command the drone's yaw angle.

The drone predominantly follows a circular path to retain a consistent horizontal distance from the Husky, effectively resetting any resultant yaw angle. Maintaining a projective distance of 6 meters between the Husky and the drone involves utilizing both the distance estimate derived from pose estimation and height data acquired from the pressure sensor. Consequently, this ensures the vertical angle between the Drone and the Husky remains constant.

To manage these dynamics, predefined threshold values were implemented for both yaw angle and distance. The drone is directed to take corrective actions when these values are exceeded. Specifically, our study set the yaw angle threshold at +5 to -5 degrees and the distance threshold at +1 to -1 meters. Commands are transmitted to the drone, instructing it on the necessary direction to correct its position whenever these threshold values are surpassed.

This work presents a vision-based hierarchical system that allows an aerial robot to follow an unmanned ground mobile platform (see Fig. 4). To this end, the images are used to estimate the relative position and heading between the UAV and the UGV. The Control Planning Module enables the UAV to maintain an almost fixed position and heading relative to the ground vehicle. For this reason, at the beginning, they both start from the almost same location with the

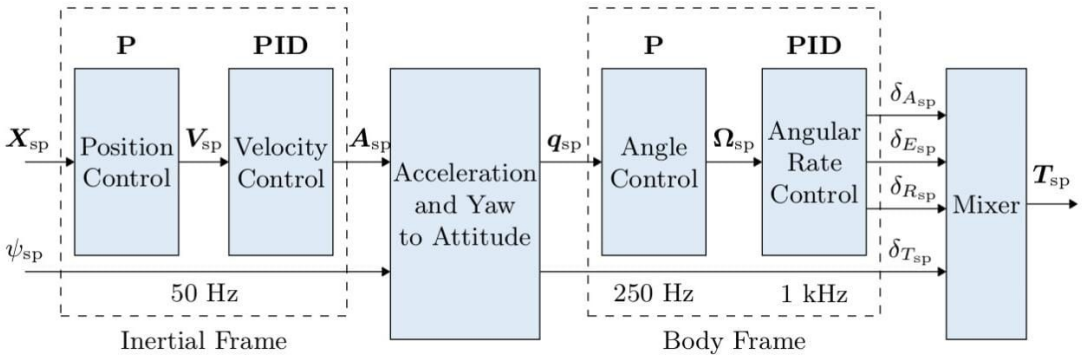

Fig. 5 UAV hierarchical control system block diagram

same orientation. The UGV is assigned as the leader, and the UAV is the follower as shown in Fig.4. The commands are generated for the UAV controller based on the UGV relative position, and relative heading values.

First, the MATLAB/Simulink software is used to construct the 6DOF flight dynamic model and the sensor models such as IMU and camera for the UAV. Second, the position and heading control algorithms are robustly developed on the real-time platform. Position control is the control of the displacement of the UAVs according to a certain reference point. The flight control algorithm based on the PX4 source code is performed on the Pixhawk board for the flight tracking position. The general control system is explained by the control loop in Fig.6, where R(s) is the input signal or the desired set point of the UAV. e(s) is the calculated positional error between the actual UAV position. u(s) is the output of the controller that alters the power of the motors. The Plant is the UAV system. C(s) is the actual position of the UAV. The Sensor is the UAV tracking system on the UGV that calculates the relative position difference between the UAV and the UGV. d(s) is the calculated distance from the sensor. The Controller is a combination of the system described in Algorithm 1 combined with the PID yaw controller. The outer-loop controller has two operation modes: autonomous and teleoperated. In autonomous mode, the estimations are fed to decoupled controllers to generate set points relative to the stabilized UAV reference frame in order to follow the UGV. This is illustrated in Fig.5. The controller provides 100 Hz feedback on the desired position, and heading of the UGV. The controller output is desired tilt angles, yaw rate and total thrust which are sent to the Pixhawk embedded controller as a reference.

The UAV utilizes the Algorithm in Fig.7 to control the x, y, and z movements as well as the yaw control of the UAV. In this algorithm, the system gets the relative position, orientation (i.e., [Δx, Δy, Δz, $\Delta\theta$]), and predetermined thresholds (i.e., [dx, dy, dz_{min}, dz_{max},d_Θ]) for these relative values as input. The outputs for each series of inputs are motor speed and propeller direction. For each input data, based on the current values of relative position, orientation, and corresponding thresholds, the UAV's motors run for a specified duration with a certain speed and direction.

In addition to the algorithm for the lateral movements, a proportional-integral—derivative (PID) control system is used to control the angular yaw movements in space. A PID control system continuously utilizes the error of the desired set point vs the actual position to calculate the motor power signal. The PID controller manages the motor power to match the UAV yaw angle with the UGV yaw angle. The controller can be expressed as Eq.10, where u is the motor control variable along a yaw axis; e is the calculated error, and the constant coefficients Kp, Ki, and Kd are tuned based upon the system.

$$u(t) = K_p e(t) + K_i \int_0^t e(t) + K_d \frac{de(t)}{dt} \tag{10}$$

After designing the controller, the PID parameters are tuned for the UAV's angular movements. In order to tune the system, an impulse of a target angle is inputted to the UAV controller, and the output of the magnetometer on the UAV is recorded to analyze the response. The PID parameters are optimized to create a responsive system without overshooting the desired target angle.

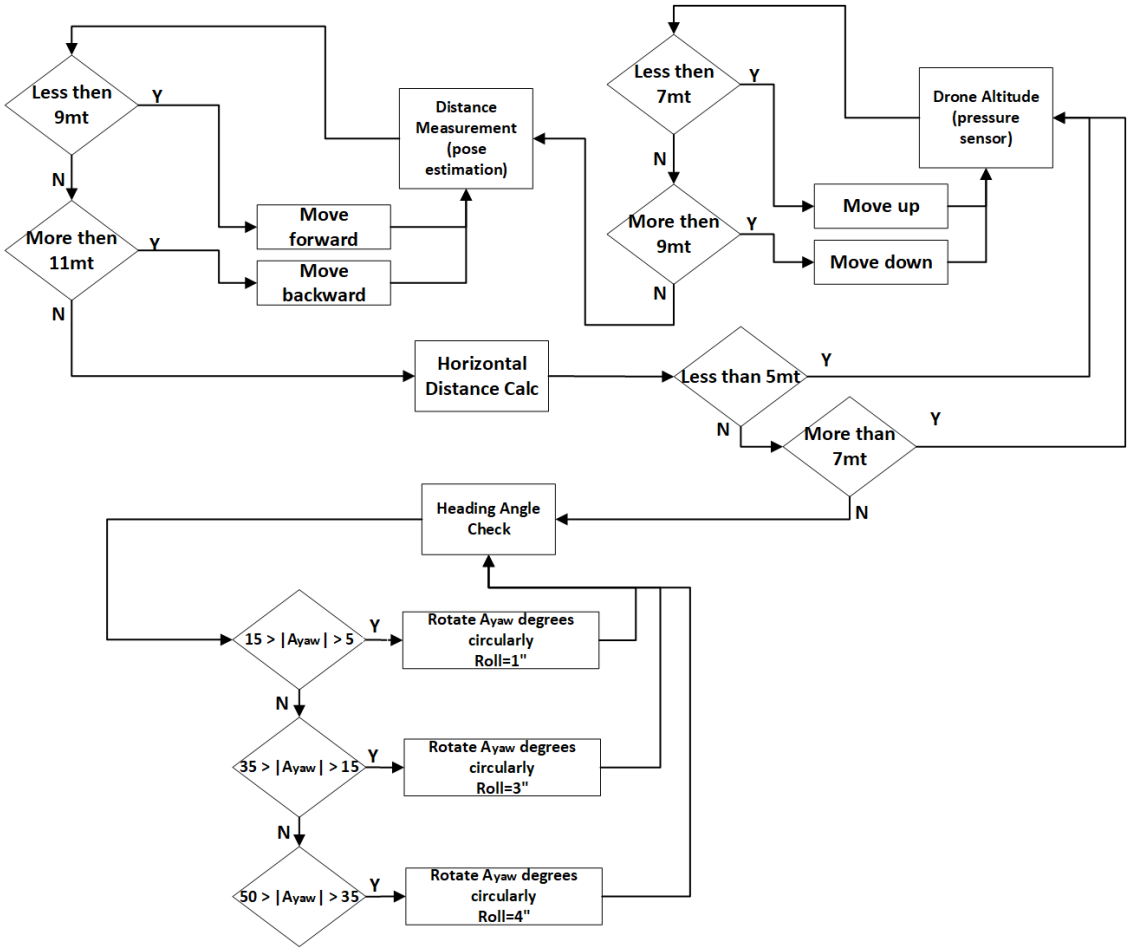

Fig. 6 General Control System for the UAV

Input:

[Δx, Δy, Δz, $\Delta\theta$]: Relative positions in X, Y, and Z axes and orientation.

[d_x, d_y, d_{zmax}, d_{zmin}, d_θ]: predetermined thresholds for relative positions and orientation

foreach received [Δx_i, Δy_i, Δz_i, $\Delta\theta_i$] **do**

  if ($|\Delta y_i| > d_y$ *or* $|\Delta\theta_i| > d_\theta$ *or* $|\Delta x| > d_x$ *or* $\Delta z_i > d_{zmax}$ *or* $\Delta z_i < d_{zmin}$ **then**

    while $\Delta y_i \neq 0$ **do**

      Rotate the blimp to the UGV location by executing a lateral motor. Move to the UGV (reducing $|\Delta y_i|$) by executing both lateral motors.

    end

    while $\Delta\theta_i \neq 0$ **do**

      Rotate the blimp to the UGV direction (reducing $\Delta\theta_i$) by executing a lateral motor.

    end

    while $\Delta x \neq 0$ **do**

      Move the blimp to the UGV (reducing $|\Delta x|$) by executing both lateral motors

    end

    while ($\Delta z_i > d_{zmax}$ *or* $\Delta z_i < d_{zmin}$) **do**

      Move the blimp vertically (satisfying $d_{zmin} < \Delta z_i < d_{zmax}$) by executing the vertical motor

    end

  end

end

Fig. 7 UAV general control algorithm[48]

IV. Result

A. UAV detection and tracking of the UGV

This section focuses on two tasks: detecting and tracking. The results showcase the best-performing detection methods tested on specific examples (validation sets). The same methods that worked best for detection are also used to assess tracking performance.

1. Dataset Creation

In this research, a custom dataset was created by collecting aerial images for the detection and tracking of unmanned ground vehicles. These images were then annotated using labeling software to identify and label objects of interest. The next step is to pre-process the aerial images to enhance their quality and extract the features. After completing the data collection, preprocessing, and annotation, we obtained a custom dataset for UGV detection and tracking. The dataset consists of a total of 1,058 annotated images, with 741 images for training, 212 images for validation, and 105 images for testing. The distribution of the dataset is as follows: 70% training, 20% validation, and 10% testing. Once the custom dataset has been created, the next step is to train object detection models using the annotated images. A data augmentation process will be also applied to the next performances of the dataset.

2. Experimental Results

This chapter presents the experimental results derived from a custom dataset for Unmanned Ground Vehicle (UGV) object detection using aerial images. The YOLOv7 deep learning model was trained and evaluated on this unique dataset. The training was conducted over 55 epochs with a batch size of 32, utilizing the Adam optimizer and a learning rate of 0.001. Performance comparison of the models was achieved using the mean average precision (mAP) and F1-score metrics on the test set. Figure 8 illustrates the evaluation results, specifically focusing on the F1-score curve and Precision curve, providing a representation of the model's performance across different thresholds. The resulting curves reflected a high level of stability and accuracy across different confidence levels. The results revealed that YOLOv7 notably outperformed, achieving the highest mAP and F1-score, registering an impressive mAP of 99.3% and an F1-score of 0.99 (see detailed evaluation results in Figure 9). Figure 10 shows some object detection results on a

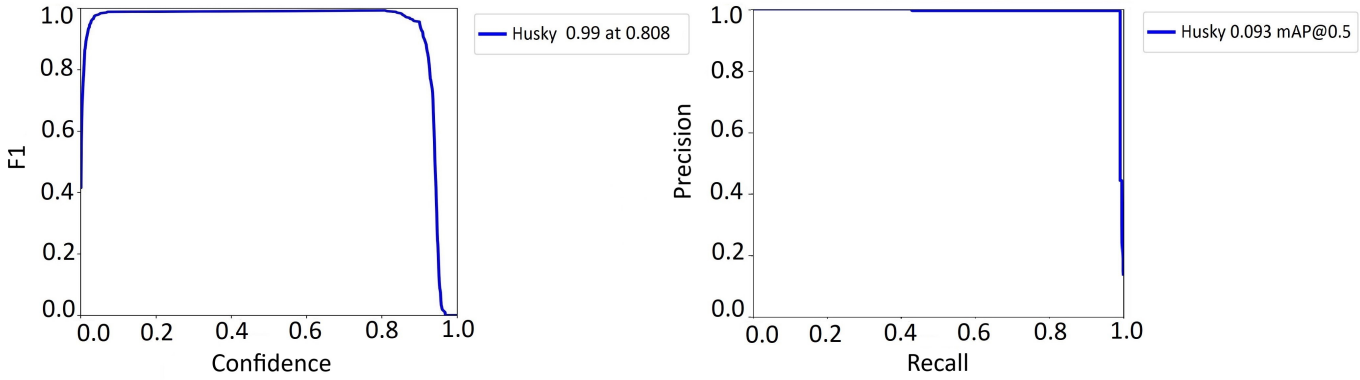

Fig. 8 The evaluation performance of the object detection model a) F1 curve and b) Precision curve

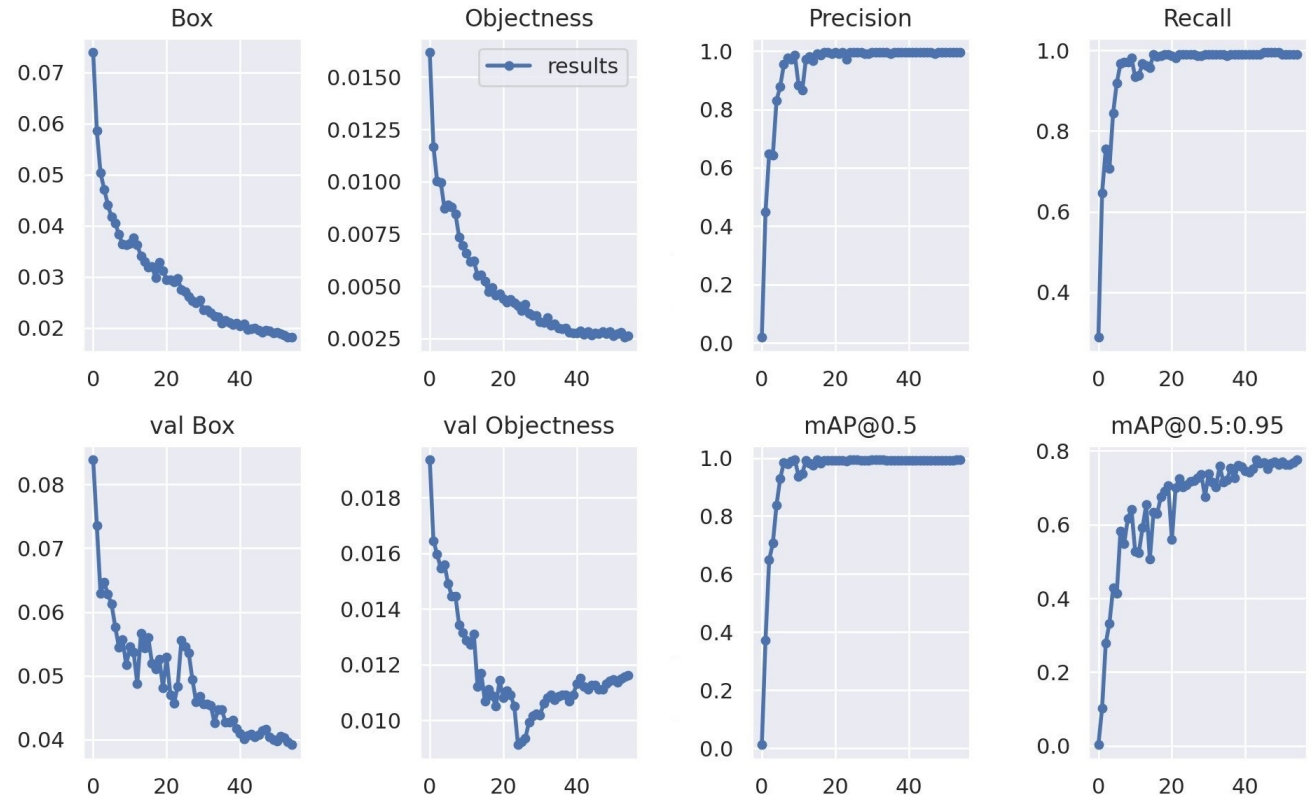

Fig. 9 The evaluation results of the object detection model

custom test set.

3. Visual-Based Pose Estimation

Building upon the successful detection and tracking of Unmanned Ground Vehicles (UGVs) using aerial imagery, this section delves into the visual-based pose estimation results. Our approach utilized the trained YOLOv7 model not only for detecting UGVs but also for estimating their poses relative to the Unmanned Aerial Vehicle (UAV). This

Fig. 10 The examples of object detection results on the custom test set

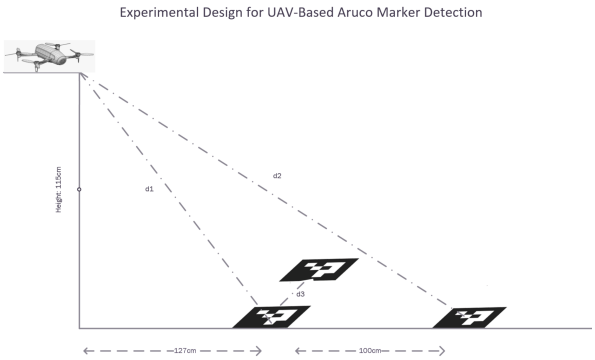

Fig. 11 Experimental design for UAV-based Aruco marker detection

Table 2 The error rate of the distances for the experimental design

	Actual Distances	Estimated Distances	Error Rate
d1	171.33cm	189cm	0.10
d2	254.47	277cm	0.089
d3	27.5cm	31.83cm	0.15

estimation is crucial for understanding the spatial orientation and positioning of UGVs in a three-dimensional space, which has significant implications for coordinated UAV-UGV operations.

The experiment was structured to verify the accuracy of our pose estimation technique. The experimental setup is illustrated in Figure 11, which shows the UAV's calibrated camera system, the positioned Aruco markers, and the specific distances measured during the experiment. We arranged Aruco markers at known distances and measured the UAV's ability to discern these distances using its calibrated vision system. Our results indicate that the visual-based system could consistently estimate distances with a high degree of accuracy, as shown in Table 2. For example, at a known distance of 175 cm (d1), the system estimated the distance with a marginal error of 0.10 cm, showcasing the precision of the calibration. Similarly, at a greater distance of 254 cm (d2), the system preserved its accuracy, with an error of only 0.08 cm. However, when measuring the horizontal distance between the markers, d3, a slight increase in error was noted at 0.15 cm. This phenomenon is well-documented in computer vision literature and can be mitigated through further refinement of the camera's intrinsic parameters or by employing more sophisticated lens models that account for such distortions.

Table 3 The yaw angle results of the relative pose estimation from a single Aruco marker

Actual Yaw Angle (degrees)	Estimated Yaw Angle (degrees)	Relative Error
0	-3	∞
15	14	0.067
30	25	0.167
60	48	0.200
90	79	0.122

In our second experiment, we explored the capabilities of a UAV in maintaining a specified distance range while following a moving UGV. The UAV, equipped with a sophisticated camera system, was tasked with capturing video frames for processing by the PoseEstimation function. This function was pivotal in detecting Aruco markers on the UGV and crucial for effective distance maintenance and orientation control. In this experiment, the PoseEstimation function plays a vital role. It divides the camera's field of view into three zones: left, central (desired), and right, corresponding to less than 25%, between 25% and 75%, and more than 75% of the frame width, respectively. The UAV's task is to keep the marker (UGV) within the central zone to ensure optimal distance maintenance. As the UGV moves, the algorithm dynamically adjusts the UAV's position. If the marker is detected in the left zone, the UAV moves rightward to recenter the marker, and vice versa as shown in the Figure 12. This zonal approach allows for nuanced control along the y-axis, ensuring the UAV adjusts its position effectively to maintain the desired distance range.

Furthermore, the algorithm calculates the UAV's orientation (yaw, pitch, and roll) using estimatePoseSingleMarkers(). These calculations are translated into Euler angles, providing precise orientation data. This data is crucial for the DroneControl function, which commands the UAV's movements, ensuring stable flight and accurate tracking of the UGV. The success of this algorithm can be demonstrated through the comparison of the algorithm's estimated yaw angles with the actual angles, as shown in Table 3. The table reveals that for a yaw angle of 0 degrees, the estimated angle was -3 degrees, indicating an infinite relative error due to the division by zero in the error calculation. At 15 degrees, the estimated angle was quite close at 14 degrees, with a relative error of 0.067. For a 30-degree yaw, the estimated was 25 degrees, with a relative error of 0.167, while at 60 degrees, the estimation fell to 48 degrees, resulting in a 0.200 relative error. The highest actual angle tested, 90 degrees, had an estimated angle of 79 degrees and a relative error of 0.122. Our study primarily focused on the principal angle region from -15 to 15 degrees. Within this range, the system displayed a tolerance for errors that did not affect performance, which is in line with our target yaw precision of 0±3 degrees. To enhance precision, we propose that augmenting the number of Aruco markers could further improve the system's accuracy. The success of the Aruco model in estimating the pose of UGVs from aerial images demonstrates the potential of integrating advanced deep-learning models with traditional computer vision techniques for UAV-UGV collaborative operations. Further work could explore the integration of additional sensory data to enhance the system's accuracy and the implementation of real-time corrective measures to compensate for the detected pose estimation errors.

B. UAV control and altitude control of UGV for Aerial Imaging

The successful implementation of UAV control mechanisms demonstrated a robust capability to maintain precise positioning relative to the UGV in off-road environments devoid of GPS signals. Analysis of visual data extracted crucial parameters such as yaw, pitch, and roll angles of the UGV, ensuring consistent updates on the distance between the Husky and the drone via pose estimation techniques as shown in the Figure 12. The coordinated circular trajectory response effectively maintained a fixed relative position between the Husky and the drone, illustrating the system's adaptability despite measured angular disparities.

In parallel, the UGV's altitude control mechanisms proved instrumental in optimizing aerial imaging quality. The collaborative utilization of data from the drone's mounted camera and additional sensors facilitated the swift creation of a comprehensive global traversability map at the ground station. This map, which classified terrain features based on navigability, significantly refined both the drone's flight path and the UGV's navigation strategies.

(a) Move Backward, Move Left

(b) Move Backward, Move Right

(c) Move Forward, Move Left

(d) Move Left

Fig. 12 Instructions Derived from Visual Outcomes in UAV-UGV Simultaneous Operation

V. Conclusion

In this study, we presented a methodology for UGV detection and tracking using aerial images acquired by a drone. The methodology involved creating a custom dataset of annotated aerial images, training and evaluating deep learning models for object detection and tracking the detected UGVs using tracking algorithms. The experimental results showed that the fine-tuned YOLOv7 model achieved the best performance on our custom dataset, with a mAP of 99.3% and F1-score of 0.99.

VI. Future Work

The future work is to integrate the detection and tracking modules into a complete system and validate its performance in real-world scenarios. The system should be able to detect and track unmanned ground vehicles in real-time and provide accurate and reliable information to the user or operator. The system should also be robust and adaptable to different environments, lighting conditions, and vehicle types. The validation can be done using different datasets and benchmarks, such as the KITTI dataset, the DAVIS benchmark, or the UAV123 dataset, which provides a variety of challenging scenarios and evaluation criteria. The validation can also involve user studies and field tests, which assess the usability and effectiveness of the system in practical applications

Acknowledgments

The authors of this paper are extremely grateful to Enver Bildik, Ebubekir Torbali, Abdurrahman Sefer Dogru, Murat Taflan and Pragadeesh Raja Shankar Narayan for his invaluable assistance and feedback in collecting aerial images and labeling the UGV data in the images. The authors are also thankful to Dr Argyrios Zolotas, and Prof Antonios Tsourdos (Centre for Autonomous and Cyberphysical Systems), for their collaboration in the flight test permissions. They all greatly assisted the research, although they may not agree with all of the interpretations/conclusions of this paper.

References

[1] Liu, D., Bao, W. K., Zhu, X., Fei, B., Xiao, Z., and Men, T., "Vision-aware air-ground cooperative target localization for UAV and UGV," *Aerospace Science and Technology*, Vol. 124, 2022. https://doi.org/10.1016/j.ast.2022.107525.

[2] Hernandez, A., Copot, C., Cerquera, J., Murcia, H., and Keyser, R. D., "Formation Control of UGVs using a UAV as Remote Vision Sensor," *IFAC Proceedings Volumes*, Vol. 47, No. 3, 2014, pp. 11872–11877. https://doi.org/https://doi.org/10.3182/20140824-6-ZA-1003.01660.

[3] Cheng, C., Li, X., Xie, L., and Li, L., "Autonomous dynamic docking of UAV based on UWB-vision in GPS-denied environment," *Journal of the Franklin Institute*, Vol. 359, No. 7, 2022, pp. 2788–2809. https://doi.org/https://doi.org/10.1016/j.jfranklin.2022.03.005, URL https://www.sciencedirect.com/science/article/pii/S0016003222001569.

[4] Gomez-Avila, J., Lopez-Franco, C., Alanis, A. Y., Arana-Daniel, N., and Lopez-Franco, M., "Ground Vehicle Tracking with a Quadrotor using Image Based Visual Servoing," *IFAC-PapersOnLine*, Vol. 51, No. 13, 2018, pp. 344–349. https://doi.org/https://doi.org/10.1016/j.ifacol.2018.07.302.

[5] Liang, X., Chen, G., Zhao, S., and Xiu, Y., "Moving target tracking method for unmanned aerial vehicle/unmanned ground vehicle heterogeneous system based on AprilTags," *Measurement and Control*, Vol. 53, No. 3-4, 2020, pp. 427–440. https://doi.org/10.1177/0020294019889074.

[6] Qwerty, O. P., "Autonomous Landing of a UAV on a Moving UGV Platform using Cooperative MPC," Master's thesis, Stockholm, Sweden, 2021.

[7] Morales, J., Castelo, I., Serra, R., Lima, P. U., and Basiri, M., "Vision-Based Autonomous Following of a Moving Platform and Landing for an Unmanned Aerial Vehicle," *Sensors*, Vol. 23, No. 829, ???? https://doi.org/10.3390/s23020829.

[8] Ghamry, K. A., Dong, Y., Kamel, M. A., and Zhang, Y., "Real-Time Autonomous Take-off, Tracking and Landing of UAV on a Moving UGV Platform," 24th Mediterranean Conference on Control and Automation (MED), Athens, Greece, 2016.

[9] Asadi, K., Suresh, A. K., Ender, A., Gotad, S., Maniyar, S., Anand, S., Noghabaei, M., Han, K., Lobaton, E., and Wu, T., "An integrated UGV-UAV system for construction site data collection," *Automation in Construction*, Vol. 12, 2020. https://doi.org/10.1016/j.autcon.2019.103068.

[10] Dufek, J., and Murphy, R., "Visual Pose Estimation of Rescue Unmanned Surface Vehicle From Unmanned Aerial System," *Frontiers in Robotics and AI*, Vol. 6, ???? https://doi.org/10.3389/frobt.2019.00042.

[11] Hood, S., Benson, K., Hamod, P., Madison, D., O'Kane, J. M., and Rekleitis, I., "Bird's eye view: Cooperative exploration by UGV and UAV," International Conference on Unmanned Aircraft Systems (ICUAS), IEEE, Miami, FL, USA, 2017. https://doi.org/10.1109/ICUAS.2017.7991513.

[12] Lange, S., Sunderhauf, N., and Protzel, P., "A vision based onboard approach for landing and position control of an autonomous multirotor UAV in GPS-denied environments," 2009 International Conference on Advanced Robotics, IEEE, Munich, Germany, 2009.

[13] Moon, H., Chellappa, R., and Rosenfeld, A., "Performance analysis of a simple vehicle detection algorithm," *Image and Vision Computing*, Vol. 20, No. 1, 2002, pp. 1–13. https://doi.org/https://doi.org/10.1016/S0262-8856(01)00059-2, URL https://www.sciencedirect.com/science/article/pii/S0262885601000592.

[14] Fefilatyev, S., and Goldgof, D., "Detection and tracking of marine vehicles in video," *2008 19th International Conference on Pattern Recognition*, 2008, pp. 1–4. https://doi.org/10.1109/ICPR.2008.4761344.

[15] Wang, H., Yu, Y., Cai, Y., Chen, X., Chen, L., and Li, Y., "Soft-Weighted-Average Ensemble Vehicle Detection Method Based on Single-Stage and Two-Stage Deep Learning Models," *IEEE Transactions on Intelligent Vehicles*, Vol. 6, No. 1, 2021, pp. 100–109. https://doi.org/10.1109/TIV.2020.3010832.

[16] Ren, S., He, K., Girshick, R. B., and Sun, J., "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks," *CoRR*, Vol. abs/1506.01497, 2015. URL http://arxiv.org/abs/1506.01497.

[17] Wang, C.-Y., Bochkovskiy, A., and Liao, H.-Y. M., "YOLOv7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors," , 2022.

[18] Zhang, Y., Sun, P., Jiang, Y., Yu, D., Weng, F., Yuan, Z., Luo, P., Liu, W., and Wang, X., "ByteTrack: Multi-object Tracking by Associating Every Detection Box," *Computer Vision – ECCV 2022*, edited by S. Avidan, G. Brostow, M. Cissé, G. M. Farinella, and T. Hassner, Springer Nature Switzerland, Cham, 2022, pp. 1–21.

[19] Hartley, R., "In defense of the eight-point algorithm," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 19, No. 6, 1997, pp. 580–593. https://doi.org/10.1109/34.601246.

[20] Mousavian, A., Anguelov, D., Flynn, J., and Kosecka, J., "3D Bounding Box Estimation Using Deep Learning and Geometry," *CoRR*, Vol. abs/1612.00496, 2016. URL http://arxiv.org/abs/1612.00496.

[21] Morales, J., Castelo, I., Serra, R., Lima, P. U., and Basiri, M., "Vision-Based Autonomous Following of a Moving Platform and Landing for an Unmanned Aerial Vehicle," *Sensors*, Vol. 23, No. 2, 2023. https://doi.org/10.3390/s23020829, URL https://www.mdpi.com/1424-8220/23/2/829.

[22] Choi, Y. C., and Ahn, H. S., "Nonlinear control of quadrotor for point tracking: Actual implementation and experimental tests," *IEEE/ASME Transactions on Mechatronics*, Vol. 20, No. 3, 2018, pp. 1179–1192. https://doi.org/10.1109/TMECH.2014.2329945.

[23] Raiesdana, S., "Control of quadrotor trajectory tracking with sliding mode control optimized by neural networks," *Proceedings of the Institution of Mechanical Engineers, Part I: Journal of Systems and Control Engineering*, Vol. 234, No. 10, 2020, pp. 1101–1119. https://doi.org/10.1177/0959651820932716.

[24] Sonugur, G., "A Review of quadrotor UAV: Control and SLAM methodologies ranging from conventional to innovative approaches," *Robotics and Autonomous Systems*, Vol. 161, 2023. https://doi.org/https://doi.org/10.1016/j.robot.2022.104342.

[25] Burggräf, P., MartínezView, A. R. P., and Wagner, P., "Quadrotors in factory applications: design and implementation of the quadrotor's P-PID cascade control system," *SN Applied Science*, Vol. 1, No. 722, 2019. https://doi.org/10.1007/s42452-019-0698-7.

[26] Salih, A. L., Moghavvemi, M., Mohamed, H. A. F., and Gaeid, K. S., "Flight PID Controller Design for a UAV Quadrotor," *Scientific Research and Essays*, Vol. 5, No. 23, 2010, pp. 3660–3667.

[27] Kuantama, E., Tarca, I., and Tarca, R., "Feedback Linearization LQR Control for Quadcopter Position Tracking," *2018 5th International Conference on Control, Decision and Information Technologies (CoDIT)*, 2018, pp. 204–209. https://doi.org/10.1109/CoDIT.2018.8394911.

[28] Setyawan, G. E., Kurniawan, W., and Gaol, A. C. L., "Linear Quadratic Regulator Controller (LQR) for AR. Drone's Safe Landing," 2019 International Conference on Sustainable Information Engineering and Technology (SIET), Lombok, Indonesia, 2019, pp. 228–233. https://doi.org/10.1109/SIET48054.2019.8986078.

[29] Reyes-Valeria, E., Enriquez-Caldera, R., Camacho, S., and Guichard, J., "LQR control for a quadrotor using unit quaternions: Modeling and simulation," 2013, pp. 172–178. https://doi.org/10.1109/CONIELECOMP.2013.6525781.

[30] Ghamry, K. A., Dong, Y., Kamel, M. A., and Zhang, Y., "Real-time autonomous take-off, tracking and landing of UAV on a moving UGV platform," *2016 24th Mediterranean conference on control and automation (MED)*, IEEE, 2016, pp. 1236–1241.

[31] Erginer, B., and Altug, E., "Design and implementation of a hybrid fuzzy logic controller for a quadrotor VTOL vehicle," *International Journal of Control, Automation and Systems*, Vol. 10, 2012, pp. 61–70. https://doi.org/https://doi.org/10.1007/s12555-012-0107-0.

[32] Castillo-Zamora, J. J., Camarillo-GóMez, K. A., PéRez-Soto, G. I., and RodríGuez-ReséNdiz, J., "Comparison of PD, PID and Sliding-Mode Position Controllers for V–Tail Quadcopter Stability," *IEEE Access*, Vol. 6, 2018, pp. 38086–38096. https://doi.org/10.1109/ACCESS.2018.2851223.

[33] Altug, E., Ostrowski, J. P., and Taylor, C. J. a., "Quadrotor control using dual camera visual feedback," IEEE International Conference on Robotics and Automation, 2003, p. 4294–4299. https://doi.org/10.1109/SIET48054.2019.8986078.

[34] Chen, F. C., Jiang, R., Zhang, K., Jiang, B., and Tao, G., "Robust Backstepping Sliding-Mode Control and Observer-Based Fault Estimation for a Quadrotor UAV," *IEEE Transactions on Industrial Electronics*, Vol. 63, No. 8, 2043, pp. 5044–5056. https://doi.org/10.1109/TIE.2016.2552151.

[35] Xu, R., and Ozguner, U., "Sliding Mode Control of a Quadrotor Helicopter," *Proceedings of the 45th IEEE Conference on Decision and Control*, IEEE, 2006, pp. 4957–4962. https://doi.org/10.1109/CDC.2006.377588.

[36] Raffo, G. V., Ortega, M. G., and Rubio, F. R., "An integral predictive/nonlinear H control structure for a quadrotor helicopter," *Automatica*, Vol. 46, No. 1, 2010, pp. 29–39. https://doi.org/https://doi.org/10.1016/j.automatica.2009.10.018.

[37] Jiang, F., Pourpanah, F., and Hao, Q., "Design, Implementation, and Evaluation of a Neural-Network-Based Quadcopter UAV System," *IEEE Transactions on Industrial Electronics*, Vol. 67, No. 3, 2020, pp. 2076–2085. https://doi.org/10.1109/TIE.2019.2905808.

[38] Gomez, V., Gomez, N., Rodas, J., Paiva, E., Saad, M., and R, G., "Pareto Optimal PID Tuning for Px4-Based Unmanned Aerial Vehicles by Using a Multi-Objective Particle Swarm Optimization Algorithm," *Aerospace*, Vol. 7, No. 6, 2020, pp. 2076–2085. https://doi.org/10.3390/aerospace7060071.

[39] George, R. P., and Prakash, V., "Real-Time Human Detection and Tracking Using Quadcopter," *Intelligent Embedded Systems*, edited by D. Thalmann, N. Subhashini, K. Mohanaprasad, and M. S. B. Murugan, Springer Singapore, Singapore, 2018, pp. 301–312.

[40] Bochkovskiy, A., Wang, C., and Liao, H. M., "YOLOv4: Optimal Speed and Accuracy of Object Detection," *CoRR*, Vol. abs/2004.10934, 2020. URL https://arxiv.org/abs/2004.10934.

[41] Redmon, J., and Farhadi, A., "YOLOv3: An Incremental Improvement," *CoRR*, Vol. abs/1804.02767, 2018. URL http://arxiv.org/abs/1804.02767.

[42] Terven, J., and Cordova-Esparza, D., "A Comprehensive Review of YOLO: From YOLOv1 and Beyond," , 2023.

[43] Cai, Z., and Vasconcelos, N., "Cascade R-CNN: High Quality Object Detection and Instance Segmentation," *CoRR*, Vol. abs/1906.09756, 2019. URL http://arxiv.org/abs/1906.09756.

[44] Zhang, Y., Sun, P., Jiang, Y., Yu, D., Yuan, Z., Luo, P., Liu, W., and Wang, X., "ByteTrack: Multi-Object Tracking by Associating Every Detection Box," *CoRR*, Vol. abs/2110.06864, 2021. URL https://arxiv.org/abs/2110.06864.

[45] Bouguet, J., "Matlab Camera Calibration Toolbox," 2000.

[46] Fetić, A., Jurić, D., and Osmanković, D., "The procedure of a camera calibration using Camera Calibration Toolbox for MATLAB," *2012 Proceedings of the 35th International Convention MIPRO*, 2012, pp. 1752–1757.

[47] Marut, A., Wojtowicz, K., and Falkowski, K., "ArUco markers pose estimation in UAV landing aid system," *2019 IEEE 5th International Workshop on Metrology for AeroSpace (MetroAeroSpace)*, 2019, pp. 261–266. https://doi.org/10.1109/MetroAeroSpace.2019.8869572.

[48] Asadi, K., Suresh, A. K., Ender, A., Gotad, S., Maniyar, S., Anand, S., Noghabaei, M., Han, K., Lobaton, E., and Wu, T., "An integrated UGV-UAV system for construction site data collection," *Automation in Construction*, Vol. 112, 2020, p. 103068.