Exprimental Unmanned Aircraft Systems: Final Report

Hao Hu, Ka Ho Lai, Xiujin Liu, Andy Qin

Abstract— **Quadrotor helicopters are emerging as a popular platform for unmanned aerial vehicle (UAV) research, due to the simplicity of their construction and maintenance, their ability to hover, and their vertical take off and landing (VTOL) capability. Motion capture (MOCAP) is used in military, entertainment, sports, medical applications, and for validation of computer vision[1] and robots[2]. MOCAP systems for quadrotor allow us to easily obtain 3D vehicle position (x,y,z) and orientation (quaternion or Euler angles) with reference to a global coordinate frame. Along with the continuous development of quadrotor and computer vision in recent years, the automatic tracking function has been gradually received more and more attention. In this project, we first developed a MOCAP algorithm to localize the quadrotor and thus provide a foundation for yaw and pitch PID control. Based on that, we then proposed a novel workflow for quadrotor to recognize, track, and follow moving people along with an innovation in structure design. Our tracking system performs well performance and stability during the real-world experiments, when human is walking, the quadrotor aircraft, facing itself towards the tracked human, can follow the human and be maintained between 2–3m ranges from the object.**

I. Introduction

A quadcopter, also called quadrocopter, or quadrotor[3], is an emerging rotorcraft concept for unmanned aerial vehicle (UAV) platforms. The small size and low inertia of drones allows use of a particularly simple flight control system, which has greatly increased the practicality of the small quadrotor in this application. The quadrotor, with four rotors and two pairs of counter-rotating, fixed-pitch blades, has two main advantages over comparable vertical take-off and landing (VTOL) UAVs, such as helicopters. First, quadrotors do not require complex propulsion systems for rotor actuation, relying instead on fixed pitch rotors and using variation in motor speed for vehicle control. This simplifies the design and reduces the cost of maintenance. Second, the use of four rotors ensures that individual rotors are smaller in diameter than the equivalent main rotor on a helicopter, relative to the airframe size. The individual rotors, therefore, store less kinetic energy during flight, mitigating the risk posed by the rotors should they make contact with any objects.[4]

Motion capture systems provide precision data that validate the accuracy of LiDAR, radar, IMU and other sensors and are helpful when refining performance in complex scenarios, such as the transition from roads to indoor surroundings. Obtaining 3D vehicle position and orientation is crucial for accurate PID control as well as other more complex tasks.

Object tracking is an important task within the field of computer vision. The proliferation of high-powered computers, the availability of high quality and inexpensive video cameras, and the increasing need for automated video analysis has generated a great deal of interest in object tracking algorithms. There are three key steps in real-time video analysis: detection of interesting moving objects, tracking of such objects from frame to frame, and analysis of object tracks to recognize their behavior. Therefore, the use of object tracking is pertinent in the tasks of motion-based recognition, automated surveillance, video indexing, human-computer interaction, traffic monitoring, and vehicle navigation.

In this project, we replace the embedded system's built-in sensors with a multi-camera motion capture system for more accurate PID control. Also, we applied a novel 3D human tracking algorithm that replies on a monocular RGB camera. we have the following key innovations and contributions :

1. we designed an innovative structure system and electronic system to integrate a video camera on our quadrotor for capturing the human pose.

2. we proposed a scene diversion pipeline to implement different algorithms based on the different contexts.

3. we generated a depth estimation algorithm to accurately estimate the depth of the point where a human is localized in 2D image, which allows us to track people using a monocular RGB image.

4. To achieve more accurate tracking in a closer scene, we combined computer vision with low-level control to implement real-time fine-tuned tracking.

II. Method

Our proposed method for achieving human tracking with the quadrotor involves maintaining a fixed altitude defined by the user. To ensure the desired altitude is maintained, we utilize a motion capture (mocap) system to provide the necessary altitude feedback to the drone. Computer vision is employed to identify the human and calculate their distance relative to the drone. The distances along the x and y axes are sent to the BeagleBone, which performs control of the yaw and pitch, ensuring that the human remains at the center of the vision system.

A. Electronic System

1) Component Selection: According to Table 1, we have integrated the following electrical components into our quadrotor in addition to the basic systems required to get it operational. These components include the BeagleBone Blue, Power Distribution Board, distributed power supply (individual LiPo Battery and portable charger), Electronic Speed Controllers, and motors.

There are several reasons why we decided to select the components mentioned above. First, the Camera Module 3 is fully supported by the Raspberry Pi platform, reducing

TABLE I
ELECTRICAL COMPONENT LIST

Item	Component	Note
1	Raspberry Pi 5	-
2	Raspberry Pi 5 Camera Module 3	-
3	Portable Charger	-
4	Motion Capture System	Outside the quadrotor
5	Xbee Receiver and Transmitter	Outside the quadrotor

the need for unnecessary configuration while achieving the goal of capturing the environment at an acceptable quality. This means the video being captured is not too noisy and includes RGB video capabilities. Additionally, the camera is very compact, which minimizes the space it occupies on the drone and ensures it does not add significant weight.

We chose the Raspberry Pi because it provides sufficient computational power to run the human tracking algorithm and supports the necessary protocols to communicate with both the camera and the BeagleBone. Furthermore, it has a wide range of well-supported libraries and extensive online documentation, making it easier to resolve any issues encountered during development.

Instead of using the LiPo battery to power the Raspberry Pi 5, we added a portable charger to the drone. The main reason for extended air time. Powering the BeagleBone, motors, Raspberry Pi 5, and its camera from the one LiPo battery would result in higher power consumption. By using a portable charger, we can reduce the time needed to recharge the LiPo battery, which is a significant advantage. Although the trade-off is the increase of the drone weight, based on our modular design concept, the drone can be quickly put into another job without carrying the vision system.

Due to the limitations of the accelerometer built into the BeagleBone, as discussed in the Low-Level Control Report, we have abandoned its use for determining the drone's altitude relative to the ground. Instead, we rely on the motion capture system, which uses Xbee to transmit and receive signals captured by external cameras, to locate the drone and provide altitude feedback.

2) Hardware Configuration: The Raspberry Pi 5 Camera Module 3 is connected to one of the 4-lane camera transceivers on the Raspberry Pi 5. The Raspberry Pi 5 communicates with the BeagleBone via UART, where the Transmit Data pin (TxD) and Receive Data pin (RxD) are connected to one of the UART ports on the BeagleBone. The Raspberry Pi 5 is powered by a portable charger. Considering that the 3.3V pin on the UART port does not provide sufficient power to operate the Raspberry Pi 5, the 3.3V pin on the UART port of the BeagleBone is not being used.

The Raspberry Pi 5 and the camera module function as a "sensor," providing information about the distance between the quadrotor and the human being tracked. Details about the algorithm used for human tracking are explained in the Human Tracking section. The calculated distance is sent to the BeagleBone via UART for control purposes.

The motion capture system, on the other hand, relies on XBee modules for data transmission and reception. One XBee serves as a ground station, transmitting x, y, z, and quaternion data to another XBee mounted on the quadrotor. For our application, only the z-axis data (altitude of the quadrotor relative to the ground) is used. Like the Raspberry Pi 5, both XBee modules communicate with the BeagleBone using UART.

3) Software Configuration: Similar to `xbee_receive.c`, we have created an additional `computerVision.c` to configure the UART and handle communication between the Raspberry Pi and BeagleBone for receiving data streamed from the Raspberry Pi. The baud rate is set to 115200 to increase the data transfer rate. Additionally, two start bytes are defined, and a checksum is included at the end to ensure the BeagleBone receives the same data content as sent by the Raspberry Pi, thereby minimizing the risk of errors.

B. Mechanical System

In addition to the prebuilt quadrotor frame designed to hold the BeagleBone, ESCs, motors, and battery, we have added a holder for the Raspberry Pi 5 and a 45-degree mount to secure the camera in place. To avoid impacting the overall structure of the quadrotor and to increase the distance between the ground and the electrical components, we have also added landing legs to the bottom for stability during landing. The exploded view of the overall structure is shown in Fig.1

The holders (Fig.2, Fig.3) and landing gear (Fig.4) are 3D-printed, enabling rapid prototyping to reduce development time. Furthermore, the material used for 3D printing is PLA, which adds minimal weight to the drone while providing a strong and durable solution.

C. Human Tracking

"UAV human tracking" typically refers to the use of Unmanned Aerial Vehicles (UAVs) or drones for tracking human movement or activities. The primary purpose of human tracking in this context is to monitor, track, or gather data on human movements in a particular area. human tracking mainly involved identifying human beings in real-time through image recognition, UAV control, and following the movement of a person in real-time.

As shown in Fig.5, In our project, we outfitted our quadrotor with a monocular RGB camera angled downward at 45 degrees. The goal is to overcome the depth limitation and perform 3D human tracking using only monocular images, all while keeping the equipment to a minimum.

We first use MobileNetV2[5], an efficient object segmentation model which pretrained on COCO dataset[6], to do the human segmentation on 2D image and return a bounding box with coordinates of four vertices $(u1, v1, u2, v2, u3, v3, u4, v4)$.

Then for 3D tracking, we mainly analyzed two different contexts and designed a scene diversion pipeline to implement different algorithm based on different context.

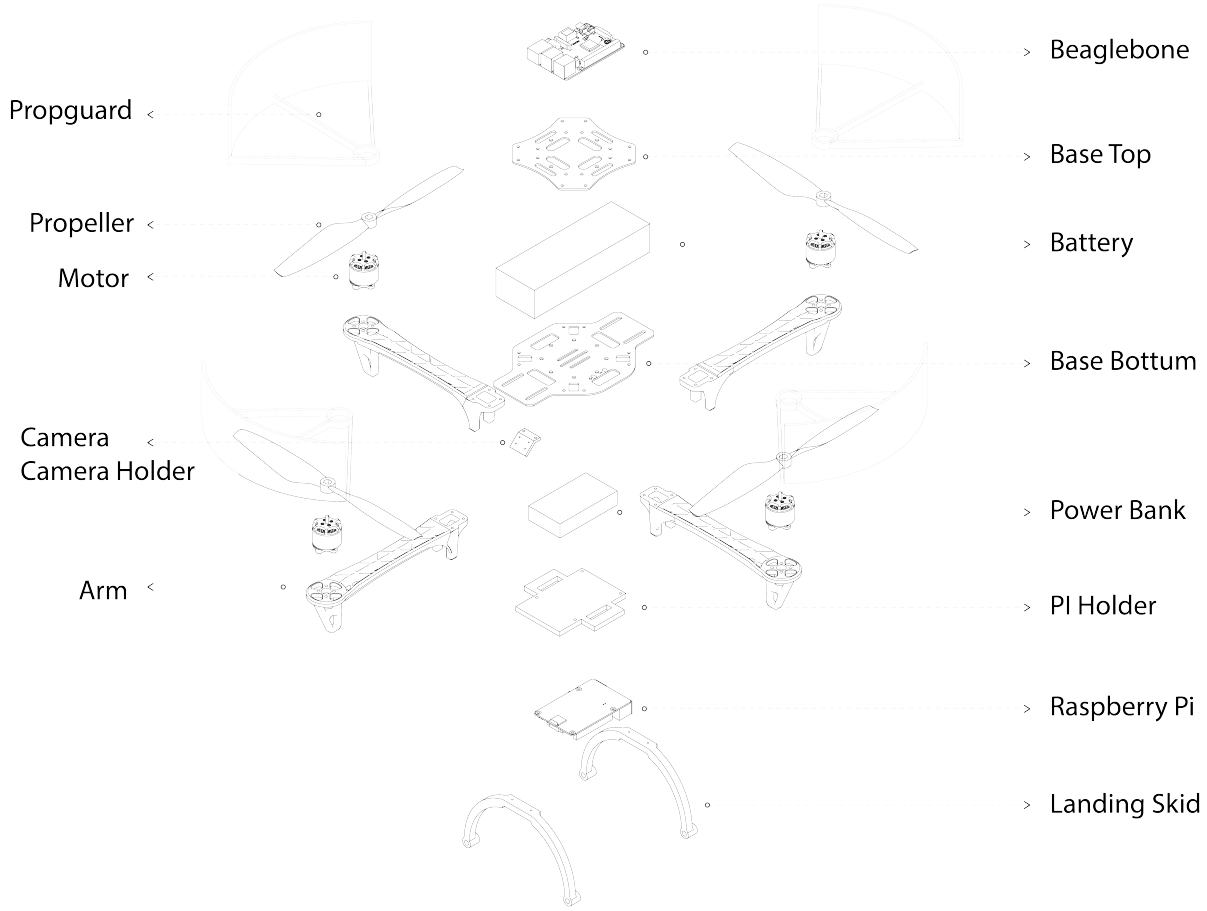

Fig. 1. Structure Exploded view

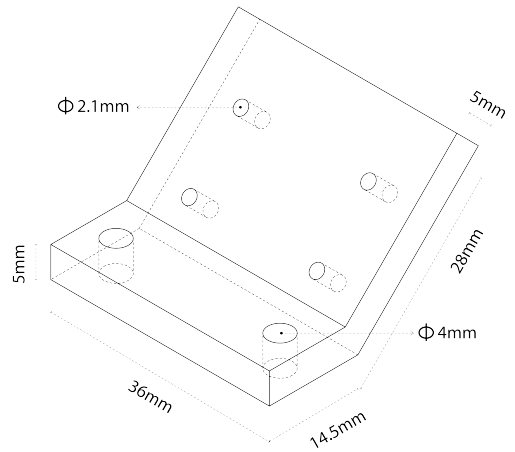

Fig. 2. Camera Holder

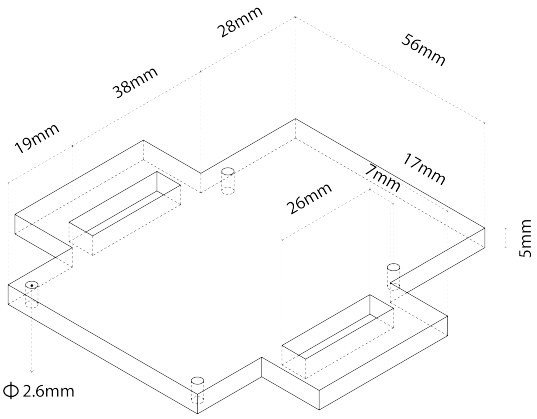

Fig. 3. Raspberry Pi Holder

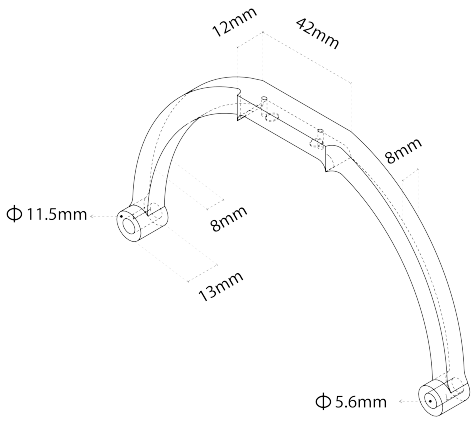

Fig. 4. Landing Skid

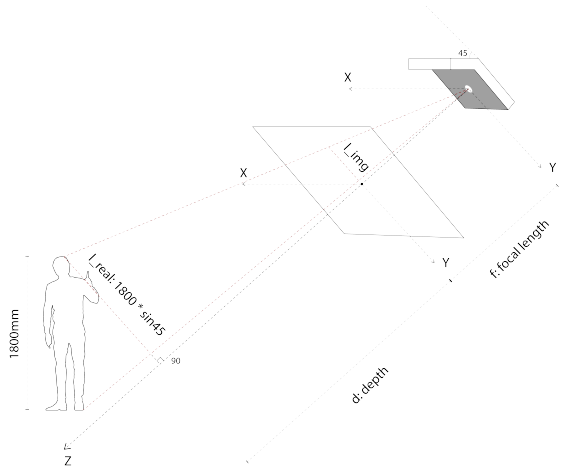

Fig. 6. Depth Estimation

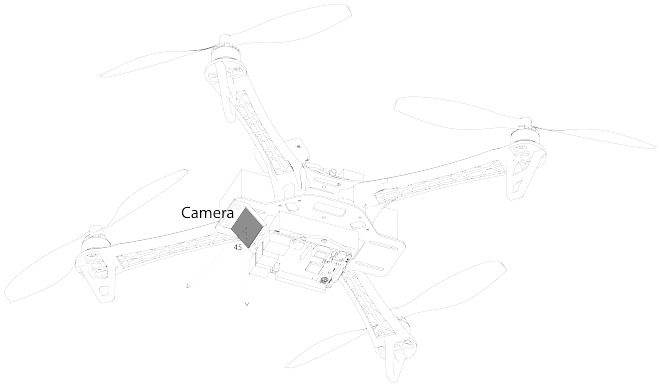

Fig. 5. Camera Position

1) Long-distance tracking: Long-distance tracking is suitable under the scenario when people is far away from the camera, and his entire body is exposed within the camera's view. This method is primarily used for rough localization of people at a distance, quickly bringing them into a specific area, with a lower emphasis on precision.

The basic idea of long-distance tracking is estimating the camera depth information, then do the coordinates transformation based on the estimated depth information to get the specific 3D location information of human, which is the guidance for UVA tracking.

As shown in Fig.6, when one's entire body is exposed within the camera's view, we assume his height is always 1800 millimeters, and always vertically stand. This means the projection of the person onto the x-y plane in the camera coordinate system is $1800 * \sin \pi/4$. Thus, we could estimate the camera depth information by using Equ.1, where d represents depth, f represents focal length, l_{real} represents the projected length of the person onto the x-y plane in the camera coordinate system, and l_{img} represents length of the person in image coordinate system, which is converted from pixel coordinate system by using Equ.2. whereu_1, v_1, u_2, v_2 is pixel coordinate of the center points of bounding box, u_0, v_0 represents the coordinates of the image coordinate system's origin O_1 in the u-v coordinate system, and d_x, d_y denote the physical dimensions of each pixel along the x and y axes, respectively. Finally, with the depth information we could get the coordinates (x, y, z) of people's location related to quadrotor coordinate system by using intrinsic and extrinsic matrix (Equ.3, Equ.4), compare with the origin of quadrotor coordinate system, we can get the movement we need to track people.

$$\frac{f}{d+f} = \frac{l_{img}}{l_{real}} \tag{1}$$

$$\begin{aligned} x_1 &= d_x * (u_1 - u_0) \\ x_2 &= d_x * (u_2 - u_0) \\ y_1 &= d_y * (v_1 - v_0) \\ y_2 &= d_y * (v_2 - v_0) \\ l_{img} &= \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2} \end{aligned} \tag{2}$$

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \frac{1}{Z_c} \begin{bmatrix} \alpha & s & u_o \\ 0 & \beta & v_o \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \mathbf{I} & | & 0 \end{bmatrix} \begin{bmatrix} X_c \\ Y_c \\ Z_c \\ 1 \end{bmatrix} \tag{3}$$

$$\begin{bmatrix} X_c \\ Y_c \\ Z_c \\ 1 \end{bmatrix} = \begin{bmatrix} & \mathbf{R} & & \mathbf{T} \\ 0 & 0 & 0 & 1 \end{bmatrix} \times \begin{bmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{bmatrix} \tag{4}$$

2) Fix-altitude Tracking: Fix-altitude Tracking utilizes MOCAP and is used under a context that human is close to quadrotor and we need a more precise real-time tracking. In this system, we use 2D image input with human bounding boxes to track and follow a human. The key concept in our tracking approach is to minimize the horizontal displacement of the human from the center of the camera frame, which

is represented as x-deviation (horizontal offset dx) and y-deviation (vertical offset dy). Our control system aims to minimize these values within a defined tolerance to ensure that the human remains centered in the frame. The example input is shown in Fig.7

The tracking process works in tandem with a fixed altitude control, which allows the drone to maintain a consistent distance from the human. This is essential for effective human-following, as the system compensates for variations in height, ensuring the drone does not inadvertently change altitude while tracking.

The control code operates in the following steps:

- Hovering when no human is detected: If no human is detected in the camera frame, the drone hovers in place, maintaining its position and altitude.
- Yaw correction for dx: Upon detecting a human, the system first adjusts the drone's yaw to correct for any horizontal offset (dx) between the bounding box center and the frame center. This keeps the human aligned along the horizontal axis.
- Pitch correction for dy: Next, the system adjusts the drone's pitch to correct for any vertical offset (dy), ensuring that the human stays centered vertically within the frame.

The advantage of this approach is its integration with the motion capture system, which provides precise positioning of the drone relative to the human. By maintaining a fixed altitude and using the bounding box to track the human, the drone is able to keep a consistent distance from the human, even as the human moves. The camera is positioned to point 45 degrees downward, providing an optimal viewing angle for detecting and tracking the human while maintaining spatial awareness of the drone's position in the environment.

In summary, the combination of fixed altitude control and bounding box-based tracking enables the drone to follow a human smoothly, with consistent distance and accurate positioning within the camera's field of view.

D. Control Scheme

The BeagleBone receives data about the deviation in distance between the tracked human and the center of the frame along the x and y axes (denoted as dx and dy, respectively), streamed from the human tracking algorithm running on the Raspberry Pi. These dx and dy deviations are further processed on the BeagleBone to determine whether to adjust the pitch or yaw setpoints while maintaining a fixed altitude. The lower-level control scheme remains unchanged and follows the implementation detailed in the Low-Level Control Report.

Our testing approach is to first evaluate each control component—altitude control, yaw control, and pitch/roll control—individually. Once each control is confirmed to meet our expectations, we will integrate all the controls for combined testing. This step-by-step approach simplifies the debugging process, making it easier to identify and resolve issues if unexpected behavior occurs. To test the individual performance of yaw and pitch/roll control, the quadrotor is

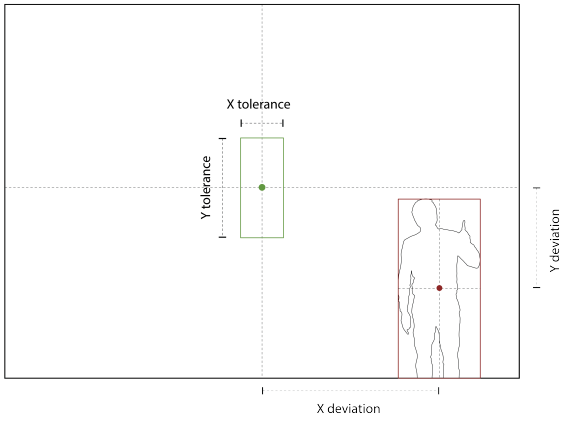

Fig. 7. Illustration of 2D Human Tracking Input

Fig. 8. Testing Demonstration for Yaw and Pitch/Roll Control (Individually)

suspended in the air by ropes, as shown in Fig.8. To test the altitude control, on the other hand, we expect the quadrotor to fly to the desire altitude automatically once it is armed.

1) Altitude Control: As shown in Fig.9, the control of altitude is entirely independent of the horizontal displacement (dx and dy) and focuses solely on the vertical axis (z). A user-defined throttle command is hard-coded to set a feedforward velocity rate for the altitude control. This feedforward velocity rate contributes to the cumulative altitude setpoint, which continues to update until it reaches the user-defined target altitude, also hard-coded.

For feedback, the quadrotor relies on a motion capture system to measure the actual altitude. The error between the altitude setpoint and the feedback is processed through a PID control block, which outputs a velocity setpoint. This velocity setpoint represents the desired rate of altitude change.

Next, the error between the velocity setpoint and the actual altitude rate (streaming from the motion capture system)

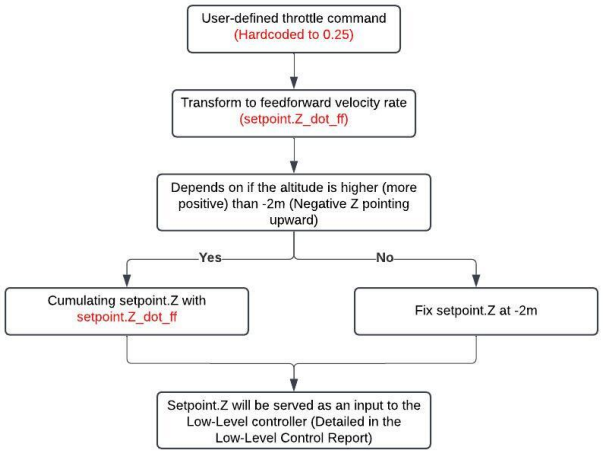

Fig. 9. Altitude Control Scheme

is processed through a second PID control block. This block computes the desired vertical acceleration input for the quadrotor.

To achieve the correct throttle command, the computed vertical acceleration is added to the base hover throttle, which ensures the minimum throttle needed to maintain hover. The total throttle command is further adjusted by dividing it by the product of the cosine of the roll and pitch angles. This adjustment ensures that the thrust continues to provide the necessary vertical force, even when the quadrotor is tilted during roll or pitch maneuvers.

With this control scheme, the quadrotor effectively tracks the altitude setpoint, maintaining stable and smooth vertical motion while compensating for tilting actions.

2) Yaw Control: As described in the Lower-Level Control Report, the yaw control focuses on controlling the yaw rate rather than directly controlling the yaw angle.

To adapt the yaw control for human tracking (shown in Fig.10), the yaw setpoint is determined based on the proximity of the tracked human to the quadrotor, specifically by checking whether the dx distances (e.g., from computer vision feedback) fall within a specific range.

If the values of dx are within (-0.05,0.05), the quadrotor maintains its current yaw angle by setting the yaw setpoint to the current yaw feedback. This effectively stops the quadrotor from yawing unnecessarily.

However, if dx falls outside this range, the yaw setpoint accumulates over time negatively or positively (depending whether the current dx is higher than 0.05 or less than -0.05, respectively) using a predefined feedforward yaw rate until dx return to the range. This cumulative behavior is similar to the altitude control scheme, where the feedforward term drives the system toward a desired setpoint gradually over time.

Once the yaw setpoint has been determined using this logic, it is compared with the state estimates provided by the motion capture system and gyro. The resulting error is processed through the low-level control system, as detailed

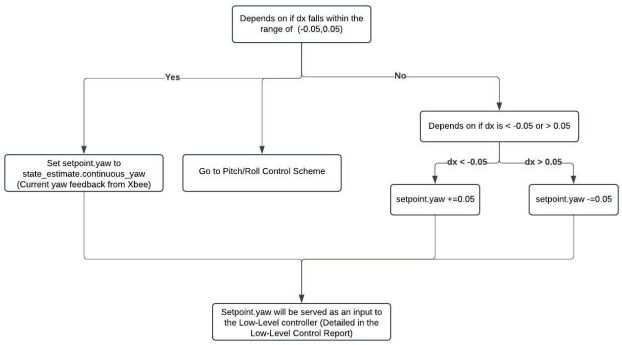

Fig. 10. Yaw Control Scheme

in the Low-Level Control Report.

3) Pitch/ Roll Control: Pitch and roll control are only activated if dx falls within the range of (-0.05,0.05). This means the quadrotor will first perform a yaw maneuver to correct its yaw position before executing any pitch or roll maneuver to bring dy within the range of (-0.05,0.05). In other words, dy will not be corrected until dx has been brought within the defined range.

As stated in the Low-Level Control Report, for both pitch and roll, the control system outputs both the angle and the angular rate. This enables the implementation of two feedback loops: an outer loop for controlling the angle and an inner loop for controlling the angular rate. The advantage of this approach is that it smooths out rapid setpoint changes caused by sudden movements of the RC remote joysticks, ensuring a continuous transition between pitch and roll setpoints and maintaining smooth, stable control.

With human tracking involved, there are some modifications to this scheme (shown in Fig.11). First, only one axis (either pitch or roll) is typically needed for human tracking, as the quadrotor can yaw to align itself with the tracked human. Second, the setpoint for pitch or roll depends on whether the tracked human's position falls within a specific range for dy (e.g., (-0.05,0.05)).

If dy is within (-0.05,0.05), the setpoint is set to the current pitch state estimate divided by 2, which provides a smooth transition to bring the pitch back to 0. This prevents unnecessary and rapid change of pitch or roll adjustments when the human is already aligned with the quadrotor. If, however, dy is outside this range, the setpoint becomes 0.1 for dy that is less than -0.05, or -0.1 when dy is greater than 0.05.

III. Result and Discussion

1) Human Tracking: The human tracking system provides the necessary dx and dy data, which are used to determine the appropriate setpoint inputs for the control system. As shown in Fig. 12, the X and Y positions in the bottom of the frame represent the dx and dy values, respectively. These values are sent to the BeagleBone to execute the logical control operations. The red dot represents the center of the bounding box surrounding the tracked human. The dx and

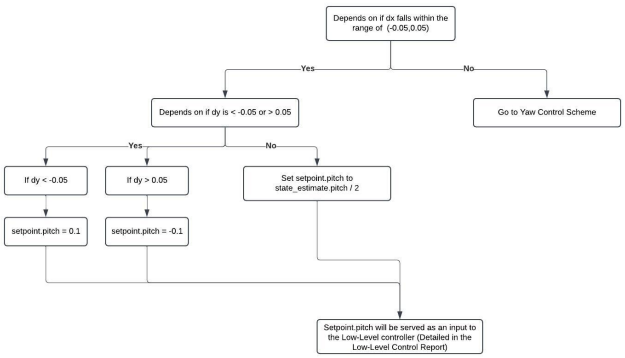

Fig. 11. Pitch/Roll Control Scheme

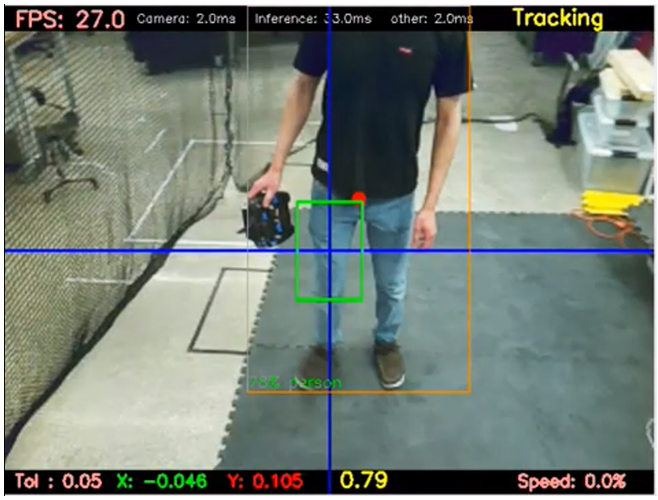

Fig. 12. Snapshot of Human Tracking frame

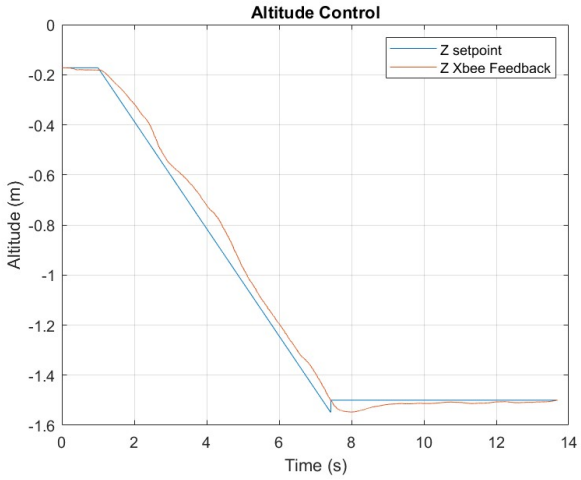

Fig. 13. Altitude Control Result without Yaw and Pitch/Roll

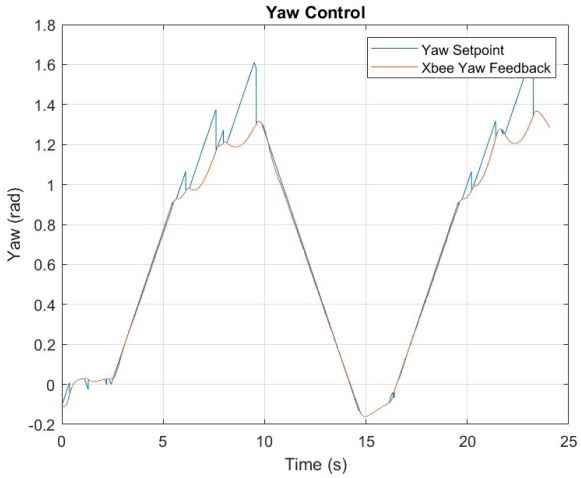

Fig. 14. Yaw Control Result without Altitude and Pitch/Roll

dy values correspond to the distance between the red dot and the center of the frame. The system stably works throughout the process.

2) Altitude Control: As shown in Fig. 13, the quadrotor demonstrates excellent performance in tracking the altitude setpoint command, with a desired height of -1.5 m (negative z-axis pointing upward from the ground). Instead of implementing a step change in the setpoint, a hardcoded feedforward velocity rate is used to contribute to the cumulative altitude setpoint. This approach generates a ramp input signal, resulting in a smooth, linear transition from the ground to the desired altitude. The gradual nature of this transition allows the control system sufficient time to respond effectively to the altitude setpoint.

It is worth noting that the initial altitude at t = 0 s is not exactly 0m, but approximately -0.17 m. This discrepancy arises from a few factors. First, reflective markers are attached to the top of the quadrotor frame. When landing skids were added, the position of the reflective markers shifted upward, causing a slight deviation in the measured height. Additionally, inaccuracies in the motion capture system, which can depend on calibration quality, may also contribute to this offset.

Despite this minor deviation, the altitude control performs exceptionally well, achieving a smooth and accurate tracking of the desired setpoint. With this success, we can now proceed to the development and testing of the yaw control.

3) Yaw Control: The yaw setpoint changes based on whether the detected human is within a preset dx range of (-0.05,0.05) or not. Similar to the altitude setpoint, the yaw setpoint accumulates over time using a fixed feedforward rate. This approach generates a smooth, linear change in the yaw setpoint, which allows the control system sufficient time to respond effectively.

As shown in Fig. 14, the quadrotor performs well in tracking the yaw setpoint. However, there are a few instances where the tracking is not perfect. We believe this discrepancy is caused by the testing conditions, where the quadrotor was suspended in the air by ropes. The ropes restrict the quadrotor's motion, which may have impacted its ability to respond accurately to the yaw setpoint.

4) Pitch Control: As seen in Fig. 15, the pitch control does not track the setpoint command effectively with the

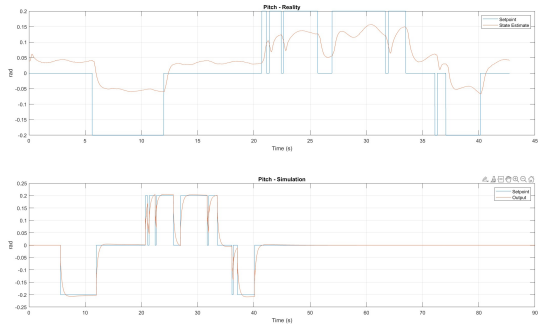

Fig. 15. Pitch Control Result without Altitude and Pitch/Roll (Top). Pitch Control Simulation Result (Bottom)

current control scheme. Several factors contribute to this behavior.

First, the motion of the quadrotor is restricted by the rope. While the control system demonstrates a tendency to track the setpoint command, the external force applied by the rope may prevent the quadrotor from achieving its desired motion.

Second, the control system's response is slower compared to the setpoint command. Unlike the yaw and altitude control schemes, where the setpoint accumulates over time using a feedforward rate, the pitch setpoint command is a fixed step value. This step setpoint changes abruptly between -0.2, 0, and 0.2 based on the condition of dy, giving the control system insufficient time to respond effectively to the rapid transitions.

As discussed in the Low-Level Control Report, the simulation closely replicates the system's behavior when using the same PID gains and input commands. Fig. 15 presents the simulation results, which do not account for nonlinear behavior or disturbances caused by the rope. These results demonstrate that the system should be able to track the input setpoint command effectively under ideal conditions.

5) Combined Altitude, Yaw, Pitch Control: Unfortunately, after combining altitude, yaw, and pitch controls, the quadrotor did not perform as expected. The altitude control successfully reached the desired height with an acceptable deviation, as shown in Fig. 16. Similarly, the yaw control determined the setpoint and executed control based on the dx distance of the tracked human, as illustrated in Fig. 17. However, the issue lies with the pitch maneuver, which failed to track the pitch setpoint command effectively. This limitation significantly impacted the quadrotor's ability to track the human accurately.

There are several possible reasons for this behavior. First, the algorithm used to determine the pitch setpoint may be a contributing factor. Instead of using a cumulative method to adjust the pitch setpoint incrementally, we implemented a fixed value (i.e., a step input) because the cumulative approach produced even worse performance in our tests. With the current algorithm, the pitch setpoint continuously adjusts whenever the human is out of range. If the setpoint value is too large, this can cause the quadrotor to "jump"

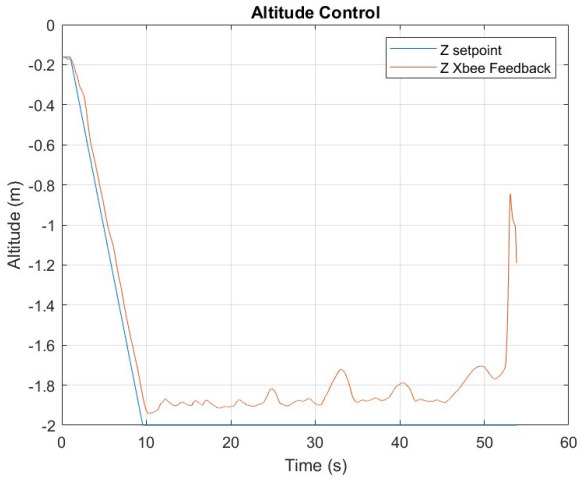

Fig. 16. Altitude Control Result with Yaw and Pitch/Roll

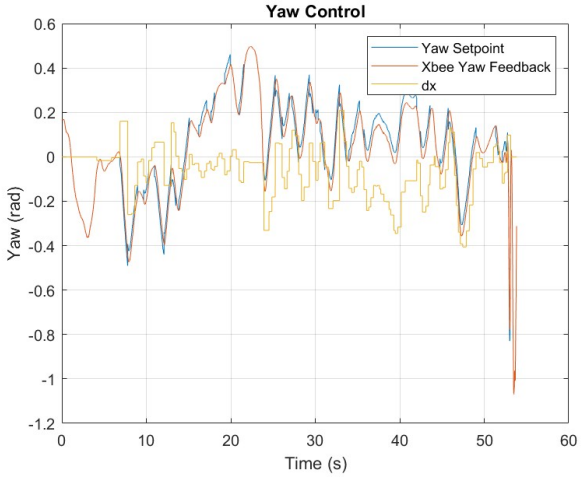

Fig. 17. Yaw Control Result with Altitude and Pitch/Roll

between the positive and negative ranges, leading to erratic behavior.

Additionally, the control system's real-world response is slower than the setpoint input changes. As a result, the system might not have enough time to respond to one setpoint command before the next command is issued, further exacerbating the "jumping" behavior. Due to the imperfections in pitch control, the quadrotor is unable to track the human stably, even though the altitude and yaw maneuvers are performing acceptably.

IV. Limitations

This project has successfully demonstrated a drone capable of human-following; however, several limitations need to be addressed to enhance its performance, reliability, and real-world applicability:

A. Altitude Drop During Forward Pitching

When the drone pitches down to move forward, its altitude often decreases unintentionally. Since the control system

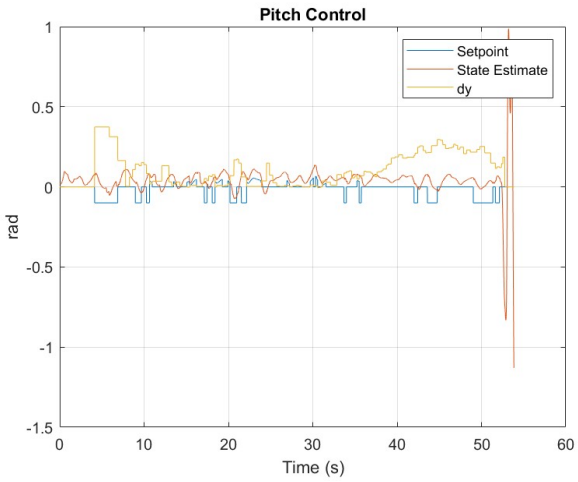

Fig. 18. Pitch Control Result with Altitude and Yaw

relies solely on 2D image inputs to track the human, this altitude drop can result in the human's position deviating from the center of the image frame. Currently, no mechanism exists to correct for this z-height variation, leading to tracking errors. A potential solution is to switch from using the human's center point to using bounding boxes. By setting constraints on how close the bounding box edges can be to the frame's boundaries, the drone can better maintain altitude and focus.

B. Abrupt Shifts in Bounding Box Center

The bounding box center for the detected person is highly dependent on the computer vision model's output and raw input data. At times, even without significant movement from the person, the bounding box center may shift abruptly, leading to sudden and erratic commands to the drone. This can cause instability and discomfort. Applying a filtering mechanism, such as a Kalman filter or exponential smoothing, can smooth out the bounding box center trajectory, ensuring more consistent drone movement.

C. Computer Vision Inference Time

The delay caused by computer vision inference can lead to outdated commands, as the drone's state may have changed by the time the command is executed. This issue is particularly problematic when the drone has a high initial speed, causing commands to misalign with the real-time scenario. To mitigate this, more computationally efficient models or hardware acceleration (e.g., edge AI accelerators) could reduce inference time, while predictive algorithms could anticipate the drone's state and adjust commands accordingly.

D. Confusion in Multi-Person Scenarios

The vision system can detect multiple individuals but struggles to consistently track a single target. This confusion increases when a person briefly exits the frame or when objects in the background are misclassified as humans. To resolve this, integrating re-identification (Re-ID) algorithms or using unique target identifiers such as color patterns or clothing features could help the drone focus on the correct individual.

E. Limited Testing Area Due to 2D Image Constraints

The current system relies solely on 2D RGB image data for tracking, which necessitates the use of a motion capture system for state estimation. This approach limits the testing area to environments equipped with motion capture capabilities. By transforming the image data into 3D coordinates, trajectory optimization could be performed to achieve more accurate and dynamic human-following behaviors without being constrained to controlled environments.

Addressing these limitations in future iterations will significantly enhance the system's adaptability, precision, and potential for deployment in real-world scenarios.

V. Conclusion

This project successfully developed a UAV system capable of tracking and following a person using monocular RGB vision and motion capture systems. The modular and structural design, with subsystem compatibility in mind, allows the system to be used in all systems that provide global positioning. The main contributions include the quadrotor structural design, the scene-based tracking pipeline, and the depth estimation algorithm, which together enable effective 3D localization and tracking of feature bodies.

The combination of altitude control, yaw, and pitch/roll mechanisms enables accurate UAV navigation, although challenges remain in achieving seamless control integration. Testing showed that altitude and yaw control performance was robust; however, the stability of the pitch response was an area for improvement. Factors such as altitude drop during forward motion, bounding box fluctuations, and real-time inference latency were identified as major limitations.

Addressing these issues through improved control algorithms, filtering techniques, and hardware optimization will enhance the system's robustness and practical applicability. By overcoming these challenges, this project lays a solid foundation for advanced UAV systems in human tracking applications, with great potential for deployment in humanitarian assistance and disaster relief scenarios.

REFERENCES

[1] D. P. Noonan, P. Mountney, D. S. Elson, A. Darzi, and G.-Z. Yang, "A stereoscopic fibroscope for camera motion and 3d depth recovery during minimally invasive surgery," in *2009 IEEE International Conference on Robotics and Automation*. IEEE, 2009, pp. 4463–4468.

[2] K. Yamane and J. Hodgins, "Simultaneous tracking and balancing of humanoid robots for imitating human motion capture data," in *2009 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2009, pp. 2510–2517.

[3] G. Hoffmann, D. G. Rajnarayan, S. L. Waslander, D. Dostal, J. S. Jang, and C. J. Tomlin, "The stanford testbed of autonomous rotorcraft for multi agent control (starmac)," in *The 23rd Digital Avionics Systems Conference (IEEE Cat. No. 04CH37576)*, vol. 2. IEEE, 2004, pp. 12–E.

[4] G. Hoffmann, H. Huang, S. Waslander, and C. Tomlin, "Quadrotor helicopter flight dynamics and control: Theory and experiment," in *AIAA guidance, navigation and control conference and exhibit*, 2007, p. 6461.

[5] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, "Mobilenetv2: Inverted residuals and linear bottlenecks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 4510–4520.

[6] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, "Microsoft coco: Common objects in context," in *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part V 13*. Springer, 2014, pp. 740–755.

[7] A. Yilmaz, O. Javed, and M. Shah, "Object tracking: A survey," *Acm computing surveys (CSUR)*, vol. 38, no. 4, pp. 13–es, 2006.