Tracking Fiducial Markers with a Drone

Daniel de Gouw
Mechatronics Engineering
University of Canterbury
Christchurch, New Zealand
dde37@uclive.ac.nz

Richard Green
Computer Science Engineering
University of Canterbury
Christchurch, New Zealand
richard.green@canterbury.ac.nz

***Abstract* – This paper proposes a simple and effective method that uses a low-cost drone for object tracking. The Parrot Mambo drone has an inbuilt Wi-Fi link that connects to an external device where the vision-based tracking and control processes are performed. A video stream from the drone provides the system with images in real-time that can be processed by the external device. A marker detection algorithm uses pre-processed images to estimate the location and orientation of fiducial markers within the drone's view. Fiducial markers are generally used in an environment to allow objects to be uniquely identified. The data from the visual tracking algorithm is used to control the drones roll, yaw and pitch angle. The algorithm ensures the marker is approximately one metre away, centred and orthogonal to the drone's view. If the tracking algorithm loses a marker, it will attempt to relocate the marker before landing.**

The proposed approach successfully detected, tracked and orientated a drone with fiducial markers. The marker detection algorithm had a reliable range of up to 7 metres and a large detection angle of 70 degrees between the axis of the marker and drone. The distance calculations had a maximum error of 10.3 percent at 7 metres. Previous research papers were found to have a lower detection range of 3 metres. In the future optical flow will be added to track markers with higher accuracy. A smoother controller will be added to the method for better efficiency.

Keywords – Autonomous Control System, Drone, Fiducial Marker, Object Tracking, Low-Cost, OpenCV, Python

I. Introduction

In recent years, there has been a significant increase in research on drones with various fields such as automated navigation, search and rescue, surveillance and photography. The operation of drones can require a lot of skill and training for the user to ensure the drone performs well for the intended application whilst preventing it from crashing. A large amount of the research papers focused on implementing artificial intelligence into drones to mitigate the issue of the user requiring comprehensive skills or training. In particular, it can be difficult for the user to maintain a constant flight distance between the drone and a moving object whilst keeping it centered in the drone's view. Currently, there are several types of drones on the market that are designed to detect and track objects autonomously, but these drones are generally expensive. The paper proposes an autonomous flight system for a low-cost drone that will track and detect a fiducial marker.

The proposed method uses a marker recognition algorithm that efficiently detects a fiducial marker and provides the markers approximate location with respect to the drone's current view. The room is scanned when no marker has been detected or if it has been lost outside the drone's view. The proposed method calculates the approximate angle of the marker from the centre of the drones view and the orientation of the marker with respect to the drone. The prior information will be used with a PID controller to manoeuvre the drone so that the marker is a specific distance away and fully aligned with the drone. The proposed method will cause the drone to maintain its distance and alignment from the marker if it is to be moved.

II. Related Work

Various amounts of research papers have been published on object tracking using drones. Object tracking techniques range from optical flow to template matching to colour detection. However, one consistent limitation was that the algorithm did not specifically orientate the drone with the detected objects so that they were square on.

III. Image Processing

One approach to detecting objects is to convert the Red Green Blue (RGB) colour space to the Hue Saturation Value (HSV) colour space [1,2,3]. Using this method, objects can be detected accurately based on their colour compared to using RGB. The RGB colour space defines how much red, green and blue is in a single value [4]. It uses an additive method where the amount of each colour determines the brightness. This colour space makes it difficult to determine how much of each primary colour a value is composed of. The HSV colour space defines the hue, saturation and value as the name suggests [5]. This allows specific colours to be detected based on the three parameters. Figure 1 illustrates both colour spaces. The HSV method uses an algorithm that performs thresholding to detect a specific colour. This thresholding can determine how well it performs with similar coloured objects and the effectiveness with different environment lighting.

Another approach is to perform template matching for detecting objects with specific geometries [6,7,8]. Efficient template matching requires the image to be converted from RGB to a grey-scale colour space. An algorithm uses the grey-scale image to identify objects with certain geometries, such as fiducial markers. These markers have a unique pattern that

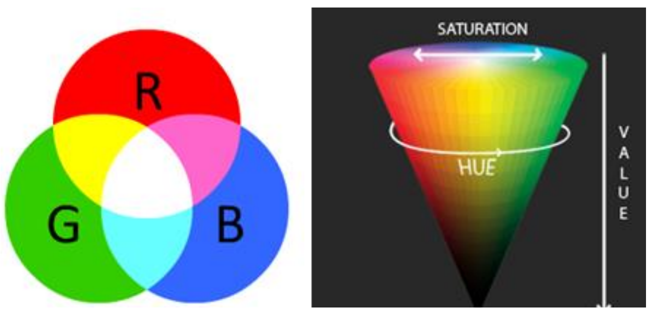

Figure 1: A comparison of the RGB and HSV colour space.

allows an algorithm to detect them efficiently, assuming the marker size and calibration parameters of the camera are known [14]. Various unique markers can be found within a dictionary that allocates each marker a unique value (ID). Initially a marker is required to be generated from a dictionary before it can be used. Figure 2 illustrates an example of a fiducial marker that has a unique ID of 24.

The image processing method highly impacts the reliability and flexibility of objects that can be detected. Converting the colour space from RGB to HSV will allow objects of a specific colour to be determined regardless of their geometry. Using a template matching algorithm allows objects with specific geometries to be detected regardless of their colour. The HSV method can also be sensitive to noise or disturbances in the environment [2]. A similar result could be seen with the fiducial markers if multiple of the exact same marker were to be used, however this is unlikely due to the large amount of unique markers.

Fiducial markers can be vulnerable to motion blur, especially when used on a drone. Various types of fiducial markers exist, with circular markers being the most resistant to motion blur [8]. However, this method does not work on real-time systems due to the long processing time of 300 milli-seconds per frame. Pose estimation is harder to perform on circular markers compared to square markers.

A. Distance Calculations

Generally, a tracking algorithm requires the distance between the marker and drone to be calculated so that a marker can be followed at a specific distance. This can be achieved through several different ways, such as using an algorithm that finds the four corners of the marker, then calculates the area and compares that to a set value [6]. This method had successful results for distances under three metres, which was a large limitation when tracking markers at unknown distances.

Another method is to use complex algorithms such as Speeded Up Robust Features (SURF) and Maximally Stable Extremal Regions (MSER) [7]. These algorithms produced improved results, but the method was very complex compared to template matching.

Another paper used a more accurate calculation to maintain a certain distance from an object [3]. This considered the objects physical width, the detected objects width in pixels, the focal length of the camera and the pixels per mm on the image sensor. This method required the specifications of the camera being used and an algorithm to determine the number of pixels the marker used. The distance calculation between the marker and the drone effects the tracking reliability. There is a trade-off between simplicity and accuracy for these methods.

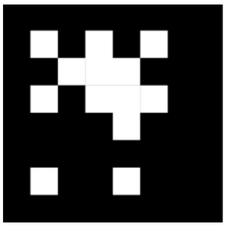

Figure 2: A 6x6 fiducial marker with an ID of 24.

B. Tracking algorithm

When tracking an object, it is important that the entire object is both centred and inside the frame. A research paper outlined the importance for keeping the object centred because it reduces the chance of it being lost outside the drone's view [6]. The algorithm determined the centre of the fiducial marker and compared this to the centre of the image. It would then try to minimise the error between the two centre points by centring the image. Figure 3 illustrates the centre of the image and marker being compared. However, the algorithm did not attempt to find the object once it had been lost. The speed at which the object could move impacted the chance of it being lost and therefore the speed must be constrained to the drone's performance capabilities [1]. If the object were to exceed the drone's capabilities, it would likely be lost outside of the drone's view and cause the detection algorithm to fail. Another consideration was that the drone couldn't see obstacles outside of its view and could easily crash if it were to hit something [3]. This paper constrained the drone to only use pitch and yaw for movement because it could crash if it were to roll left or right.

IV. Method

A. Drone and Hardware

The Parrot Mambo with an FPV camera is a small, robust and inexpensive drone, which can be seen in Figure 4. The drone encompasses a 3-axis gyroscope & accelerometer, an inertial measurement unit, a first-person view (FPV) camera, an ultrasound sensor, a pressure sensor, Bluetooth and Wi-Fi connectivity. The camera has a resolution of 640x360 with a frame rate of 30fps and a viewing angle of 120 degrees. The drone can be controlled on a device via Wi-Fi or Bluetooth and has a flight time of about 8 – 10 minutes. The drone is powered with an 800MHz ARM A9 processor.

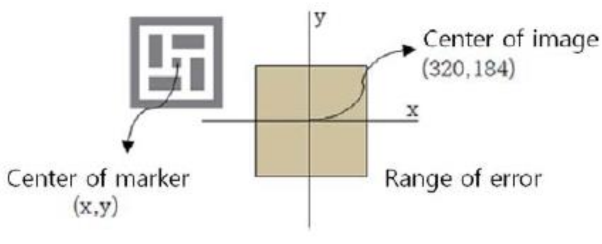

Figure 3: A comparison technique used to calculate the error between the co-ordinates of the marker and centre of the frame.

Figure 4: A Parrot Mambo drone with an FPV camera located on top.

A laptop with a Linux operating system will be used to execute the algorithm and communicate with the drone. The laptop has an Intel Core i7-4750HQ 64-BIT processor running at 2.00 GHz with 16 GB of RAM.

B. *Software*

The algorithm will be programmed in Python 3 through the Wing IDE (Version 7.0.1). The Pyparrot interface (Version 1.5.3) will be used to control the drone and receive data from its sensors [9]. This interface allows the drone to be moved in a specific direction through one simple command. The drone continuously streams real-time video from the FPV camera to the connected device via a Real Time Streaming Protocol (RTSP). This enables the laptop to receive the real-time video stream through FFmpeg.

Open CV is an open source computer vision and machine learning library for Python [10]. There are more than 2500 libraries that can be used for multiple applications such as identifying, classifying and tracking objects. OpenCV (Version 3.4.6) allows autonomous image processing in real time through many different methods. One method uses the ArUco library to detect ArUco (fiducial) markers using dictionaries that contain various unique fiducial markers [11].

The proposed algorithm in this paper consists of image conversion, marker detection, data processing and a PID controller. The algorithm relies on several iterations to successfully align with a marker. The duration of each loop is dependent on the video streams framerate and the delay in the control loop. A simplified version of the algorithm is illustrated by a flowchart in Figure 5.

C. *Image Processing*

The real-time video from the FPV camera needs to be modified before marker detection can occur. FFmpeg is used to obtain the real-time video in a separate thread from the main algorithm to avoid issues by minimising delays in receiving the stream. This allows the main algorithm to be run simultaneously, where it captures the frames from the video stream. The number of frames that can be captured is dependent on the camera's frame rate, which means a maximum of 30 frames can be captured per second. This is because capturing at a higher rate causes duplicates of frames and capturing at a lower rate causes frames to be lost. Each frame is then processed when they have been captured, causing the algorithm to be delayed if they are processed faster than they are received.

A grey-scaling and template matching technique will be implemented for the detection of fiducial markers. This was chosen over the HSV colour space approach to simplify the algorithm and prevent the drone from tracking the wrong object in an unknown environment. The HSV approach will not allow pose estimation to be performed easily and thus the orientation of an object would be hard to find. Whereas, pose estimation can be performed on fiducial markers. The drone is designed to be flown indoors making it less likely to be affected by environmental noise and disturbances. Due to this, very little motion blur is observed in the real-time stream.

Each frame from the drone will be converted from the BGR colour space to the grey-scale colour space to improve the efficiency of the template matching algorithm. This is

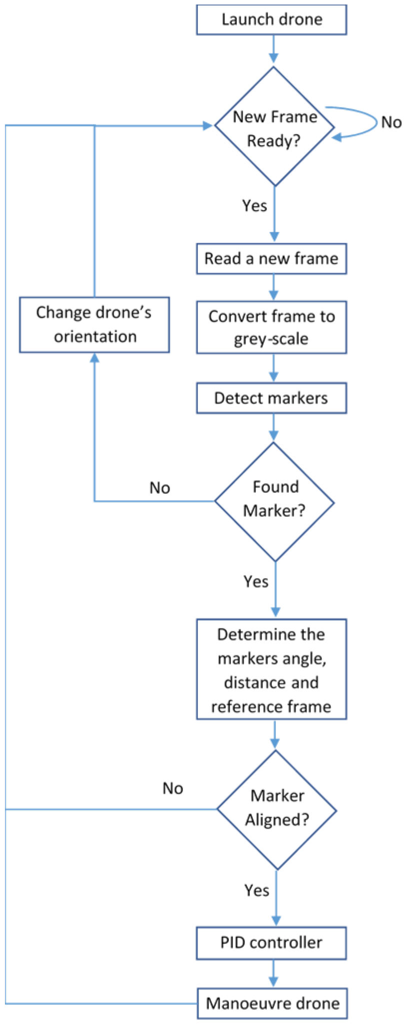

Figure 5: A flowchart of the proposed algorithm for detecting and aligning with fiducial markers.

achieved through a luminosity algorithm that extracts the RGB values of each pixel in an image and coverts them into a grey-scale value. This algorithm uses Equation 1 which weights the values of red, green and blue differently due to the human eye not perceiving colours equally [12]. The template matching algorithm has two main stages [11]. The first stage uses an algorithm that looks for potential markers, which is achieved through three steps. Firstly, adaptive thresholding is applied, then contours are determined, and some filtering is applied to improve the accuracy. The second stage looks at each marker candidate and determines whether they are a fiducial marker and what their unique ID is. The template matching algorithm returns the co-ordinates of the four corners for each marker that was detected. It will make an array for the corners of each marker from the specified dictionary, another array for the corresponding marker ID's and an array for the markers that were not in the dictionary.

Pose estimation is performed on the detected marker to determine the location of the marker and its orientation with respect to the drone. This algorithm provides a simple approach that does not require specific information to be known about the camera. The pose estimation algorithm performs a Direct Linear Transform (DLT) on the detected

$$Grey = 0.299R + 0.587G + 0.114B \qquad [1]$$

marker followed by the Levenberg-Marquardt optimization to minimize the error [13]. The DLT will calculate the translational and rotational vectors for the marker. The translational vector describes the location of the marker in the camera co-ordinate system, where the Z axis defines the distance between the marker and drone. The rotational vector defines the markers co-ordinate system with respect to the cameras co-ordinate system. The reliability and accuracy of the algorithm is improved by using markers that are printed out at a larger scale.

D. Tracking Strategy

The tracking algorithm requires the translational vector of the marker to determine where the marker is within the drone's view. The translation vector can be used to calculate the angle from the markers centre point to the cameras centre point through the equation illustrated in Figure 6. The algorithm will calculate this angle every time a new frame is received. A PID controller is used for manoeuvring the drone to minimise the error between the markers current and desired location. Figure 7 illustrates how the PID controller is used within the control loop. The PID controller outputs the percentage for the motor speeds to minimise the error. Thus, the PID controller will be used to centre the marker within the drones view and position the drone one metre away from the marker.

The tracking algorithm uses the rotational vector to determine the orientation of the marker with respect to the drone. The pose estimation algorithm gives a Rodrigues rotational vector, but an Euler matrix is required to simplify the calculations [14]. The Rodrigues vector is converted into an Euler matrix through an RQ decomposition. This provides the Euler angles about the X, Y and Z axis for the marker. The angle about the Y axis shows the change in yaw of the markers co-ordinate system with respect to the drone's co-ordinate system. Figure 8 illustrates a possible scenario where the

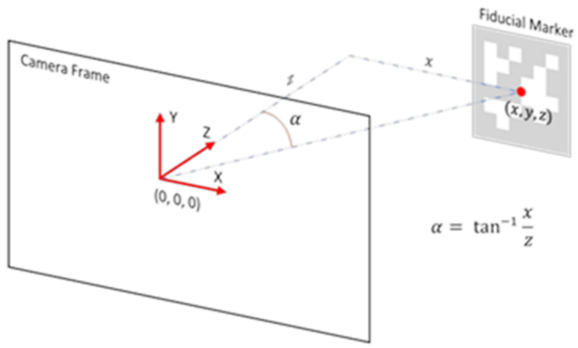

Figure 6: A diagram showing how the co-ordinate system is used to calculate the marker's angle with respect to the drone's centre of view.

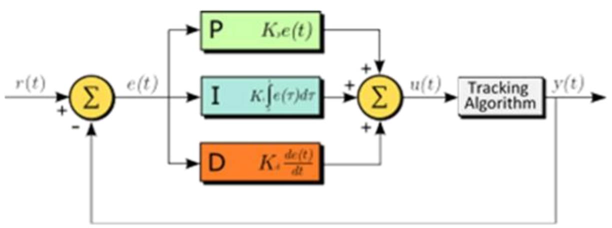

Figure 7: A diagram showing how the PID controller is used with the tracking algorithm.

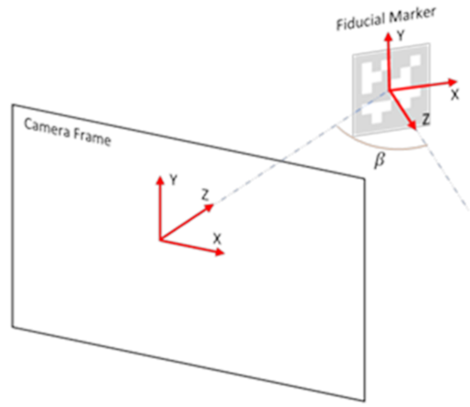

Figure 8: Illustrates the difference in angle, β degrees, between the drone and marker reference frame.

difference in reference frame angles about the Y axis is displayed as β. The tracking algorithm would spin the drone around in a circular motion using roll and yaw control to minimise this angle. This was done to prevent the distance and position of the marker within its frame from changing as the drone moved. This section of the tracking algorithm assumes the drone is being flown in a large open room with no obstacles it can crash into. This is because the drone has not been designed to detect obstacles outside of its view. An algorithm that achieves the same result without using roll control is more complex to implement.

The algorithm will be constantly centring the marker in the frame, correcting the distance from the marker and correcting the rotation between the marker and drone. This will cause the drone to follow the marker where ever it is moved. The markers are assumed to be moving slow to prevent them from escaping the drones view. If the marker is lost outside of its view, it will spin around 360 degrees to attempt to relocate the marker and land if it is unsuccessful. The same process occurs when the drone is started up.

V. Results

Several tests were performed on the proposed method to validate its accuracy and reliability. The algorithm was used with and without a marker to test the reliability of the real-time video stream. Unfortunately, the frame rate was extremely low which caused the entire system to react slowly to changes in its environment. This enforced the need for the drone and marker to move at slow speeds to reduce the number of frames that were potentially missed. The frame rate of the stream was 1.56 FPS with no marker detected and 1.15 FPS when a marker was detected. The frame rate dropped 26.3% when a marker was detected in the frame, which meant the proposed method only obtained 3.8% of the 30 FPS provided by the RTSP stream.

In Figure 9, we can see the proposed method is able to track the fiducial marker in various different cases. In the top left corner, the tracking algorithm outputs if certain parts of alignment have been achieved and the controllers next intentions. The fiducial marker was printed out to have a physical dimension of 16 by 16 centimetres. This allowed the detection algorithm to detect the marker up to a maximum of 9 metres away. However, the algorithm was only reliable for distances up to 7 metres away. This was a 233% increase in

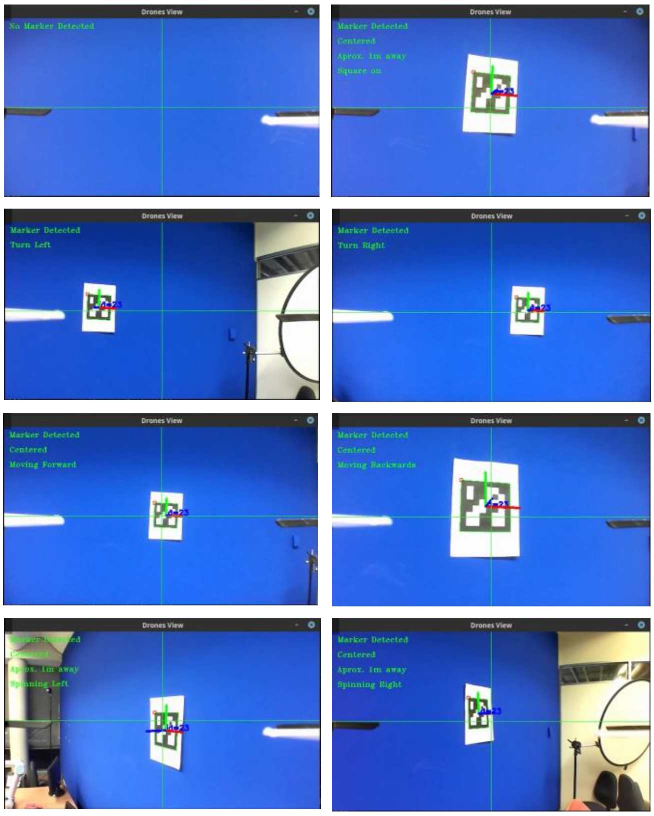

Figure 9: The drone's point of view for 8 different cases with a debug output of what the tracking algorithm is currently doing.

distance than what a prior research paper could reliably detect fiducial markers. Distances between 7 and 9 meters were unreliable due to the algorithm not having a 100% success rate at detecting the marker in every frame. The results of the calculated distance and their error from the real distance can be seen in Table 1. The error of the distance calculation increased as the distance got larger. The results show that the proposed method is optimised for small distances due to a very low error of less than 10 percent. The error on large distances was not important since the algorithm was only required to detect that the marker was too far away from the object. Therefore, the drone would fly towards the marker, decreasing the distance and improving the accuracy. The tracking algorithm positioned the drone so that the distance from the marker was between 0.9 and 1.1 metres. An error of 10% either side of the desired distance was implemented to prevent the drone from overshooting the desired position due to the low frame rate from the video stream.

Table 1: The calculated distance from the pose estimation algorithm and its error when compared to the real distance.

Calculated Distance	Real Distance	Error (%)
1.03	1.00	3.0
2.05	2.00	2.5
3.15	3.00	5.0
4.3	4.00	7.5
5.4	5.00	8.0
6.53	6.00	8.8
7.73	7.00	10.3

The angle from the marker centre point to the centre of the frame inherited the error from the distance calculation. The algorithm detected the marker on a maximum angle of ± 15 degrees, which was 25% of the cameras ± 60 degree viewing angle. The significantly reduced angle effected the drone's capabilities at tracking objects and could have been due to the frames being sized wrong by the FFmpeg operations. Some of the reduced angle was a result of the detection algorithm requiring the entire marker to be in the drones view for detection. However when the algorithm detected a marker, the drone successfully centered the marker within its view. The tracking algorithm positioned the drone so that the angle of the markers centre point to the frames centre point was between ± 2 degrees. The drone had greater precision in adjusting yaw compared to adjusting the pitch or roll. This was likely due to the limitations of the flight controller inside of the drone. The proposed method successfully detected and orientated itself with markers at a maximum orientation of ± 70 degrees from the drone. The large detection angle significantly increased the reliability of the proposed method. The tracking algorithm would reduce the angle to within ± 20 degrees.

The tracking algorithm was restricted to moving in set patterns when following and orientating itself with the marker. Therefore, it could not move forwards whilst centring the marker in its view. This reduced the efficiency of the tracking algorithm and occasionally caused some previous alignment to be lost. This was most commonly seen when the tracking algorithm was orientating the drone with markers on an angle about its Y axis. This movement caused the algorithm to realign the distance and position of the marker within the drones view to change. The thresholds for aligning the marker caused the drone to correct them as soon as they were not satisfied. This caused small movements when close to the thresholds, where the proposed method would appear frozen in the worst-case scenario. A similar scenario was observed with environmental disturbances whilst the drone was flying. Objects located near the drone or slight breezes on the drone caused the drone to wobble and change its location. An upper and lower threshold was implemented to prevent these issues. The algorithm would only reposition the drone if the distance between the drone and marker was less than 0.6 metres or greater than 1.4 metres. Similarly, the algorithm would only reposition the drone if the angle of the markers centre point from the centre of the frame exceeded 4 degrees either side of the frames centre point.

The tracking algorithm failed when a marker was used in the wrong orientation. This was because the marker detection algorithm had a fixed set of axes that were placed on the marker based on its geometry. A marker on its side would cause the Y and X axis to swap, preventing the markers orientation angle with respect to the drone to fail. The marker detection algorithm was not very accurate with its scaling parameter for the distance. Thus, the algorithm was calibrated so that it was more accurate when calculating the distance.

A PID controller was used to smoothly control the movement of the drone. The controller used two separate sets of gains for moving the drone to the desired distance and for positioning the marker in the centre of its view. The integral and derivate gains for both sets were set to a value of 1, as they were found to have very little effect due to the slow loop speeds. The proportional gain for the distance movement was set to 5. The proportional gain for the centring movement was set to 15.

VI. Conclusion

The proposed method successfully detected, followed and orientated a Parrot Mambo drone with fiducial markers. A template matching approach [6,7,8] was used instead of a HSV colour space approach [1,2,3] due to prioritising reliability over flexibility. A HSV technique is prone to false detections with similar coloured objects, and can be prone to noise or disturbances in the environment due to incorrect thresholding. Changes in lighting can have a large effect on the values for HSV. A template matching technique improves on these issues making the proposed approach more reliable. The use of fiducial markers allows the drone to be used in an unknown environment since a fiducial marker can be attached to any object, assuming the constraints are met for the drone's capabilities. It is unlikely that another identical fiducial marker is present in the drone's environment due to the uniqueness of each marker. The drone will be able to detect and differentiate between multiple markers.

The fiducial markers were printed at a larger scale so that they could be reliably detected at distances up to 7 metres away, whereas past research papers could only reliably detect markers up to 3 metres away [6]. This improved the flexibility and performance of the system due to the increased range at which the markers could be within. A method that used SURF and MSER algorithm to accurately calculate distance [7] was not chosen due to its complexity. Instead, a simple pose estimation algorithm is performed to approximately calculate the distance between the marker and drone.

The method failed at operating efficiently in real-time due to the low framerate from the drone's video stream. The tracking algorithm was limited to set movement patterns for aligning the drone with the marker. This was an inefficient approach and occasionally caused previous alignment to be lost.

A. *Future Work*

To improve the efficiency of the proposed method, a 30-fps video could be used to remove the delays in the algorithm and reduce the chance of the marker being lost outside the drone's view. This could be achieved through using VLC instead of FFmpeg since VLC is known to work but has higher complexity. Another method to achieve this would be to create a third thread for the drone commands so the main algorithms are not impacted by the delays in commanding the drone. The template matching technique will work for any object assuming it has a fiducial marker attached. Thus, the proposed method could be further improved by having the capability to select specific objects that don't have fiducial markers or by changing the object detection method. This could be achieved through using a tracking-learning-detection (TLD) algorithm that allows the user to select an object they want to track [15].

The set commands for manoeuvring the drone could be replaced with a single command that takes three inputs for the three different axes. This would provide more control on the drone and allow the drone to be aligned in all directions at once. Three separate PID controllers could be used to finetune each direction. Modifying the orientation manoeuvre to use PID control would prevent alignment from getting lost. The tracking algorithm could be further improved by adding in control for the altitude so that it can fly at a specific height with respect to the marker.

A Kalman filter and IMU could be added to the system to achieve better performance [16]. An IMU could provide feedback on the drone's movements to increase the accuracy of the tracking algorithm, with the Kalman filter ameliorating any noisy measurements. Another form of navigation, such as SLAM, could be added to improve the drone's awareness of the environment [17]. This paper outlines the need for a closed loop system to remove the drift that can occur in the control of the drone over long distances.

VII. References

[1] It Nun Thiang, Dr.LuMaw, Hla Myo Tun. "Vision-Based Object Tracking Algorithm With AR.Drone". International Journal of Scientific & Technology, Volume 5, Issue 06, pp. 135-139 June 2016.

[2] Chi-Tinh Dang, Hoang-The Pham, Thanh-Binh Pham and Nguyen-Vu Truong, "Vision based ground object tracking using AR.Drone quadrotor," International Conference on Control, Automation and Information Sciences (ICCAIS), Nha Trang, 2013, pp. 146-151.

[3] Ioannis Dramouskas, Isidoros Perikos, and Ioannis Hatzilygeroudis. "A Method for Performing Efficient Real-Time Object Tracing for Drones". Advances in Smart Systems Research, Vol. 6, pp. 55-64, 2017.

[4] "HSV vs. RGB". Internet: https://handmap.github.io/hsv-vs-rgb/, 26 Oct 2016, [Accessed 5/05/19]

[5] "The HSV Color Model in Graphic Design". Internet: https://www.lifewire.com/what-is-hsv-in-design-1078068, April 26, 2019, [Accessed 5/05/19]

[6] Jin Kim, Yoon Suk Lee, Sang Su Han, Sang Ho Kim, Gun Hyuk Lee, Ho Jun Ji, Hye Ji Choi, Kwang Nam Choi. "Autonomous flight system using marker recognition on drone." 2015 21st Korea-Japan Joint Workshop on Frontiers of Computer Vision (FCV), Mokpo, 2015, pp. 1-4.

[7] Wai Nwe Tun, Maxim Tyan, Sangho Kim, and Jae-Woo Lee. "Marker Tracking with AR.Drone for Visual-based Navigation Using SURF and MSER Algorithms". KSAS Spring Conference, 2017, pp. 2-3.

[8] Meghshyam G. Prasad, Sharat Chandran, Michael S. Brown. "A Motion Blur Resilient Fiducial For Quadcopter Imaging". IEEE Winter Conference on Applications of Computer Vision, 2015, pp. 254-261.

[9] "Pyparrot 1.5.3 documentation". Internet: https://pyparrot.readthedocs.io/en/latest/index.html, [Accessed 8/04/19]

[10] "About OpenCV". Internet: https://opencv.org/about.html, [Accessed 27/03/19]

[11] "Detection of ArUco Markers". Internet: https://docs.opencv.org/3.1.0/d5/dae/tutorial_aruco_detection.html, [Accessed 27/03/19]

[12] "Miscellaneous Image Transformations". Internet: https://docs.opencv.org/2.4/modules/imgproc/doc/miscellaneous_transformations.html#cvtcolor, [Accessed 6/05/19]

[13] Satya Mallick. "Head Pose Estimation using OpenCV and Dlib". Ineternet: https://www.learnopencv.com/head-pose-estimation-using-opencv-and-dlib/, September 26, 2016, [Accessed 30/03/19]

[14] Andrej Babinec, Ladislav Jurišica, Peter Hubinský, František Duchoň. "Visual Localization of Mobile Robot Using Artificial Markers". Procedia Engineering, Volume 96, 2014, Pages 1-9.

[15] R. Barták and A. Vykovský, "Any Object Tracking and Following by a Flying Drone," 2015 Fourteenth Mexican International Conference on Artificial Intelligence (MICAI), Cuernavaca, 2015, pp. 35-41.

[16] M. F. Sani and G. Karimian, "Automatic navigation and landing of an indoor AR. drone quadrotor using ArUco marker and inertial sensors". 2017 International Conference on Computer and Drone Applications (IConDA), Kuching, 2017, pp. 102-107.

[17] K. K. Lekkala and V. K. Mittal, "Simultaneous aerial vehicle localization and Human tracking," 2016 IEEE Region 10 Conference (TENCON), Singapore, 2016, pp.