

See discussions, stats, and author profiles for this publication at: <https://www.researchgate.net/publication/326661865>

# Visual-GPS combined 'follow-me' tracking for selfie drones

Article in *Advanced Robotics* · July 2018

DOI: 10.1080/01691864.2018.1501278

CITATIONS

0

READS

346

2 authors:



**Thanh Tuan Do**

Seoul National University of Science and Technology

3 PUBLICATIONS 1 CITATION

[SEE PROFILE](#)



**Heejune Ahn**

Skolkovo Institute of Science and Technology

55 PUBLICATIONS 737 CITATIONS

[SEE PROFILE](#)

Some of the authors of this publication are also working on these related projects:



Robot Software [View project](#)



Embedded System [View project](#)




## Visual-GPS combined 'follow-me' tracking for selfie drones

T. Tuan Do & Heejune Ahn

To cite this article: T. Tuan Do & Heejune Ahn (2018): Visual-GPS combined 'follow-me' tracking for selfie drones, Advanced Robotics, DOI: [10.1080/01691864.2018.1501278](https://doi.org/10.1080/01691864.2018.1501278)

To link to this article: <https://doi.org/10.1080/01691864.2018.1501278>




View supplementary material 



Published online: 27 Jul 2018.



Submit your article to this journal 




View Crossmark data 

FULL PAPER



# Visual-GPS combined ‘follow-me’ tracking for selfie drones

T. Tuan Do and Heejune Ahn 

Department of Electrical and Information Engineering, Seoul National University of Science and Technology, Seoul, Republic of Korea

## ABSTRACT

The ‘follow-me’ mode, where the drone autonomously follows and captures the user or target, is a new and attractive feature for camera drones, especially selfie drones. For this purpose, today’s commercial drones use the difference between two GPS data in the drone and user-side mobile GCS, e.g. a smartphone, but the targeting performance is often not satisfactory due to the inaccuracy of the GPS data, ranging from a few to tens of meters. Visual tracking can be considered for a solution to this problem, but the reliability of visual tracking is still questionable for long-term tracking in unexpected operating environments. The paper proposes a hybrid approach that combines the high accuracy of a visual tracking algorithm in short-term tracking and the reliability of GPS-based one in long-term tracking. The experiment with our prototype drone system demonstrates that the proposed combined approach can accomplish the follow-me operation very successfully, capturing the target in the center of video contents over 50% higher accuracy than the GPS-based ones. Also, the extreme scenario experiments verify the system can recover vision tracking failure and Wi-Fi failure quickly in a short-term, e.g. 3–7 s.

## ARTICLE HISTORY

Revised 20 October 2017  
Revised 28 January 2018, 22  
May 2018, and 24 June 2018  
Accepted 2 July 2018

## KEYWORDS

Selfie drone; follow-me  
mode; sensor fusion; visual  
tracking; GPS

## 1. Introduction

Recently, small or micro-UAV (unmanned air vehicle) drones have become more popular with hobbyists [1–3], though drones have been around for years extensively in the military domain. The recent popularity of drones comes from diverse reasons. First of all, a number of open-source platforms and in-depth technical information for small drones are now available from generous contributions of RC hobbyists, universities, and corporations [2,3]. At present, a bare minimal drone system can be assembled at the cost of less than 500 USD, and camera-equipped medium-grade drones are still less than 1500 USD [2]. On top of that, a drone has flexible workspace compared with traditional mobile robot system, and thus the practical application area of drones is increasing rapidly in various fields, such as remote-sensing, rescue, and delivery service [1–3].

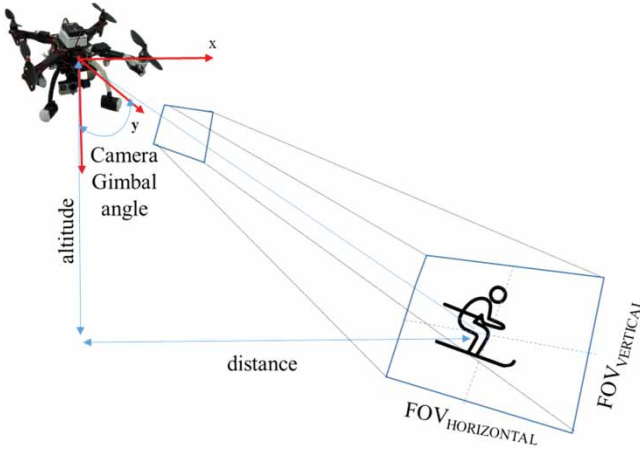
Despite the possibility of more serious usage in future applications, today’s most popular one is using drones for filming or photography [1]. Sports enthusiasts in particular – are ravenous for photos and videos taken by aerial robots. However, the current camera drone needs skilled operators for tracking and shooting the target users, which arises a big burden of dependency and inconvenience to the users. Therefore, the idea of

the autonomous mode of capturing aerial videos while the user is skiing or snowboarding surfaced, coined as ‘follow-me’ mode. Many commercial drones such as Air-dog, Solo, and Hexo+ started releasing follow-me mode enabled drones for the consumers [1].

As illustrated in Figure 1, when flying in follow-me mode, the drone automatically follows the user and captures video of the user from a pre-configured distance and angle. The actual implementation of follow-me is target tracking or drawing a ‘Lease’ line to drone. The main performance measure of quality video content is determined by the proximity of the target from the center of the video frame at the user-preferred size. Technically, we require controlling the relative drone position and the camera angle so that the camera can stare at the target position. At present, this control function is implemented using GPS data of the drone and user-side mobile GCS (ground control station). However, the limited precision of satellite navigation makes it hard to keep the drone and camera positioned perfectly [1,4]. The details are described with a mathematical model in Section 3. The researchers in [4] tried to increase the accuracy of relative GPS and achieved sub-meter accuracy in their experimental configuration. But besides its complexity of system construction and computational

**CONTACT** Heejune Ahn  [heejune@seoultech.ac.kr](mailto:heejune@seoultech.ac.kr)

 Supplemental data for this article can be accessed at <https://doi.org/10.1080/01691864.2018.1501278>



**Figure 1.** Follow-me operation conceptual model.

load, the precise relative GPS methods needs raw-GPS data from sensors and it is not available in most sensor products and GCS systems, especially in today's smartphone.

Many solutions to this problem are under research, but the industry expects real-time video processing will help, which is why drone companies have been working to outfit their drones with microcomputers powerful enough to process live video [1]. Visual tracking has been studied for several decades, and recently there has been remarkable progress in visual object tracking algorithm [5–7]. There are several notable studies [8–10] on object tracking. Recently, several studies [11,12] applied visual object tracking for drone applications, but still, the reliability of any state-of-art algorithm does not reach the requirement of the commercial 'long-term' follow-me application. The algorithms were tested only with a small search area and short-term videos. Moreover, the systems do not have a reliable way to detect and recover from tracking failures by themselves. For an instance, once the moving target gets occluded and moves out of camera coverage, no current visual tracking algorithm can redetect it. Very recently, a few companies added vision-based tracking function, especially DJI's 'active Track'. However, they do not disclose the technical information on the vision algorithm and only provide some video clips that demonstrate the visual tracking application. One YouTube video [13] shows that DJI active Track outperforms TLD and CT (compression tracking). However, there are also many YouTube videos of tracking failure of DJI's 'active Track' [14]. These videos reveal that DJI active tracking does not integrate our mode switching technique yet.

In this paper, we propose an innovative practical approach that combines the accuracy of visual tracking and reliability of GPS-based tracking. The GPS data are less accurate in compare to visual tracking localization,

but far more robust. Therefore, we use GPS data for monitoring the reliability of visual tracking algorithms. When the visual tracking results are out of confidence range of GPS data, the drone switches to GPS-based tracking mode till the visual tracker redetects the target. Our major evaluation criterion is the accuracy comparison between GPS-only mode and the proposed combined mode and visual tracking failure detection and recovery delay.

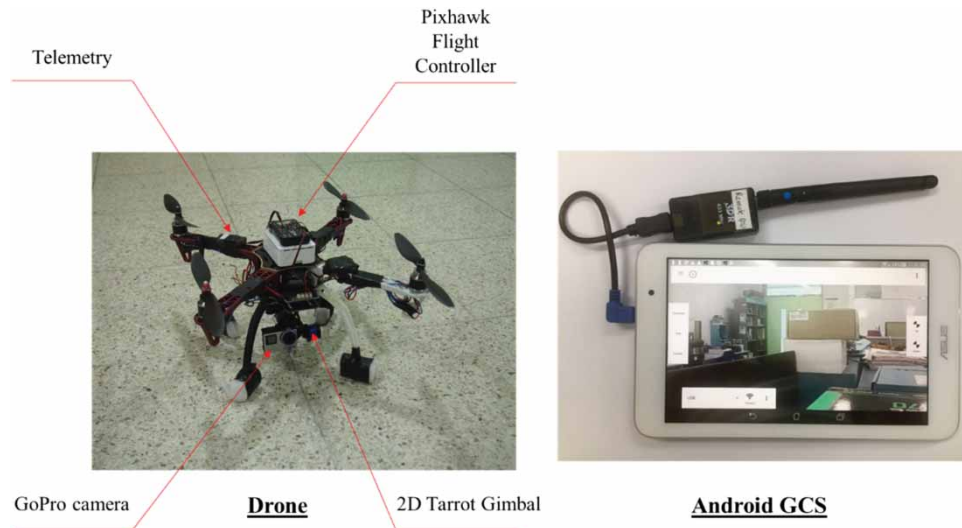
The paper is organized as follows: Section 2 introduces our drone system with a focus on follow-me operation, especially how GPS and visual tracking mode are combined. The control principle and system architecture of quadrotor drones are not covered in this paper, because there are already comprehensive literatures [2,3] and other paper in this special issue might discuss on it. Section 3 describes the details of the conventional GPS-based follow-me algorithm and their performance model for video capturing accuracy. Section 4 gives the detailed visual tracking algorithm, i.e. how to control the drone position and camera angle. In Section 5, we present the scenario-based experiments demonstrating the accuracy improvement and reliability using the combined algorithm. Section 5 concludes the paper with contribution, limitations, and future work of the paper.

## 2. Follow-me tracking system design

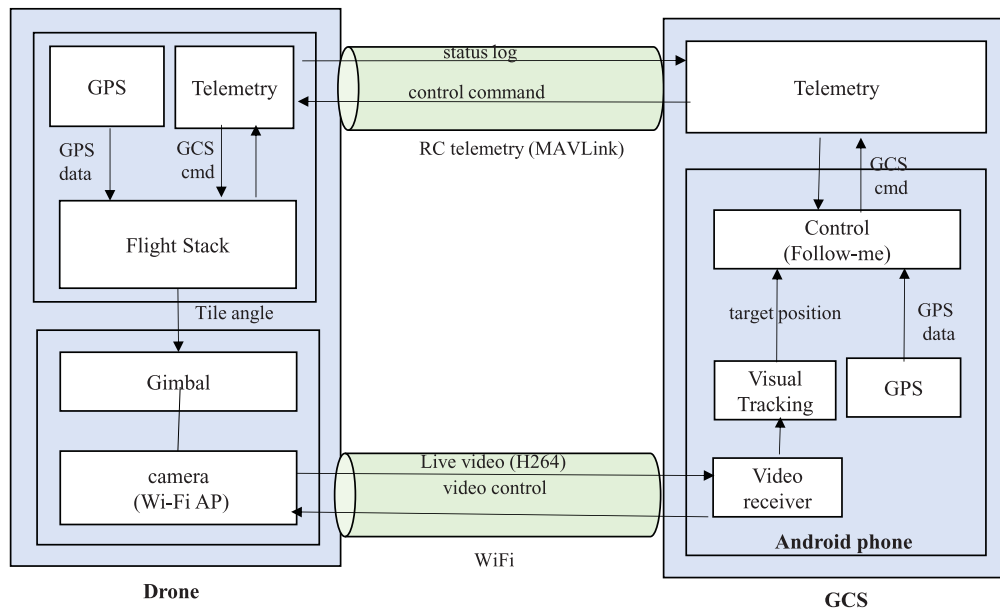
### 2.1. System overview

Figures 2 and 3 show our drone system and its software architecture for follow-me operation. The drone consists of a Wi-Fi action-camera mounted on a quadrotor drone, and Android GCS. The system configuration is similar to the existing selfie drones such as DJI, 3DR Solo, Hexo+, and Air-dog. The GCS, carried by the drone user, receives drone's status information, video stream from the camera, and also helps to control the drone's position and camera's tilt angle.

Our drone system is built based on open-source APM flight stack with Pixhawk avionics board [2]. The GCS communicates with the drone in MAVlink (Micro Air Vehicle Communication) protocol [15] over 900 MHz telemetry to receive IMU (Inertial Measurement Unit) data, GPS sensor data and other required status information of drone like velocity, direction, etc. Control commands for the drone are fed back the other way around through the same channel. MAVlink communication suffices the data transfer for GPS-based follow-me mode, while the Visual Tracking Algorithm requires more bandwidth than this header only messaging protocol. We have utilized the UDP/Wi-Fi connectivity of GoPro action-cam to pick up a low resolution live video stream of



**Figure 2.** Our prototype drone and GCS system.



**Figure 3.** Follow-me drone system structure: a drone and a mobile GCS.

424 × 240@25 fps. The GoPro camera is mounted on a 2D gimbal (tilt and pan controllable gimbal) to stabilize the videography. In our system, the gimbal only controls pan angle for stabilization in the horizontal direction, and the targeting algorithm controls the tilt angle vertically for adjusting the camera direction to the user through a MAVlink command through the drone controller. The video data from GoPro are used as input for tracking algorithm and processed using FFmpeg 3.3 (otherwise known as 'libav') library and OpenCV for Android 3.2 SDK on Google Android platform.

Target tracking and remote drone control algorithms are the major components of the Tracking Function.

For visual target tracking, Tracking–Learning–Detection (TLD) algorithm [8] was chosen because of TLD's high accuracy and capability of detecting target loss, which is hardly ever supported by other tracking algorithms [6–8]. To keep the target in the center of the captured video, the GCS controls the drone position and gimbal angle combining its own GPS data, drone's GPS data and target bounding box. In case the target is lost, GCS utilizes lost condition and combines the both side GPS information to track back the target tuning drone position and gimbal angle. Note that, the low resolution live video stream (@25 fps) are used only for user preview for visual tracking, the high-quality video is stored at drone side to source ultimate video content. Additionally, we passed

every fifth frames alone to the visual tracking algorithm to reduce the computational load.

## 2.2. Tracking modes

The procedure to start follow-me mode is as follows:

- (1) Establish the telemetry and Wi-Fi connection to the quadcopter.
- (2) Send take off command to the quadcopter take off with a certain hovering altitude.
- (3) Adjust the drone position and gimbal rotation till live camera content shows the target user.
- (4) Select an initial target bounding box around the user.
- (5) Activate follow-me mode.

The main task of tracking algorithm is to keep the target at the center as closely as possible and maintain the size and distance to the target as desired. As illustrated in Figure 4, the tracking algorithm runs in two modes: vision-based tracking mode and GPS tracking mode. The visual tracking mode is the default mode in tracking algorithm to attain higher accuracy in locating the target. When operating in visual tracking mode, the bounding box output of TLD is trained to locate and estimate the size of the target. On target loss, the algorithm switches to the GPS tracking mode. The transition between vision tracking mode and GPS tracking mode can be categorized into these following two cases:

- Visual tracking loss case 1: The visual tracking algorithm reports its failure in tracking for a period (e.g. 5–6 s). It includes the Wi-Fi connectivity disruption, as well. For the current system configuration, the waiting time of 6 s is chosen for visual tracking algorithm failure based on our experiments and following reasoning. GoPro video input can be blur or deform in a few seconds (2–3 s) when Wi-Fi connectivity is not strong enough to send video data for visual tracking algorithm. And the TLD algorithm counts

several frames for declaring the loss of target. Considering these delay components, we choose 5–6 s for the waiting delay to switch the tracking to GPS mode. In a different system and communication environment, one can decrease the switching time.

- Visual tracking loss case 2: The visual tracking algorithm reports its success in tracking but the distance between drone and user mismatches with GPS-based calculation; that delineates wrong the object is being tracked. To determine the mismatch, we compare the distance between drone and user that is estimated by using vision data ( $d_{\text{vision}}$ ) and the distance between drone and user by GPS data ( $d_{\text{gps}}$ ) which is calculated from GPS data of drone and GCS:

$$\text{If } d_{\text{gps}} > d_{\text{vision}} + \text{threshold} \Rightarrow \text{Vision is detecting wrong object.}$$

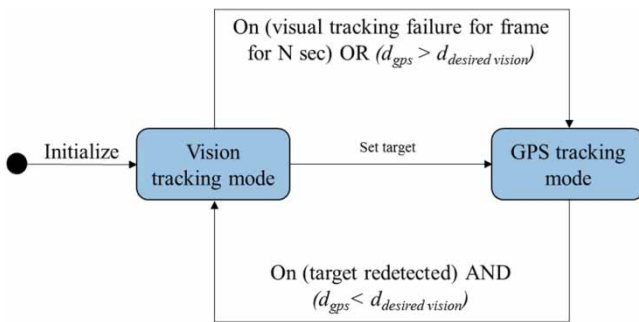
The threshold should be chosen to make sure that the mode is switched robust enough to recover vision tracking mode quickly. In Section 3.3, the experiment proves that GPS error is often more than 5 m. Also if the error distance between  $d_{\text{gps}}$  and  $d_{\text{vision}}$  is larger than 5 m, the user is out of video frames. To keep the user in the inside of video frames:

$$(d_{\text{gps}} \pm 5) - d_{\text{vision}} \leq 5.$$

In this experiment, we chose a threshold value is 10 m. For shot, we denote:

$$d_{\text{desired vision}} = d_{\text{vision}} + \text{threshold}.$$

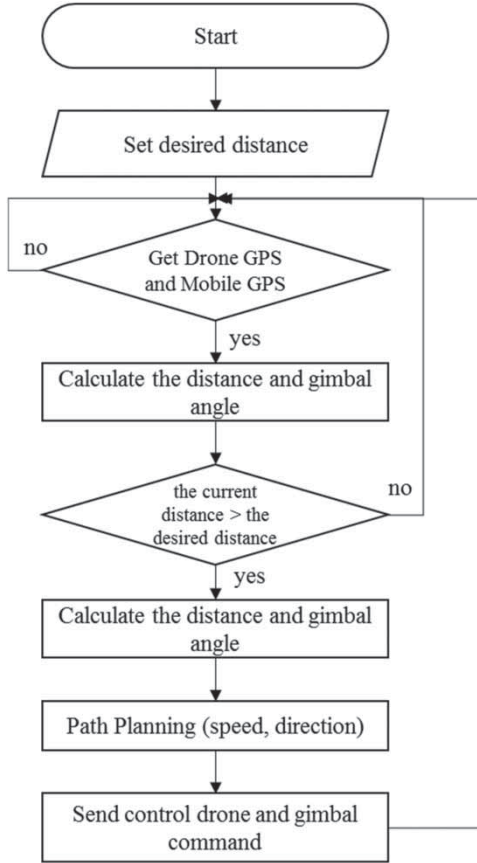
TLD [8] framework is designed for long-term tracking of an unknown object in a video stream. It combines tracking, detection, and machine learning algorithm to improve the tracking performance. Basically, the tracker and detector can track or locate the object themselves independently. But the tracker is cannot recover if the object moves out of the camera and reappears again. On the other hand, the detector considers every frame is independent, it performs a global scan on the image to localize all object appearance that has been observed and learned in the past. When the tracker and detector are used concurrently, they can complement each other by feeding back potential adjustments. The detector can re-initialize a tracker to minimize the tracking failures while the tracker can provide weakly labeled training data for the detector. This coupling improves the detector performance during run-time. Learning block of TLD observes the performance of tracker and detector. It estimates detector's errors and generates training examples to reduce errors in the future. Altogether TLD algorithm can redetect the target object when the target reappears



**Figure 4.** Visual-GPS combined camera tracking modes and transition.



in late frames in the video stream. Therefore, the camera tracking mode is switched back to visual tracking mode when TLD restarts reporting the bounding boxes of the target when the target is located within the range of GPS data tolerance.



**Figure 5.** GPS tracking mode operation in GCS.

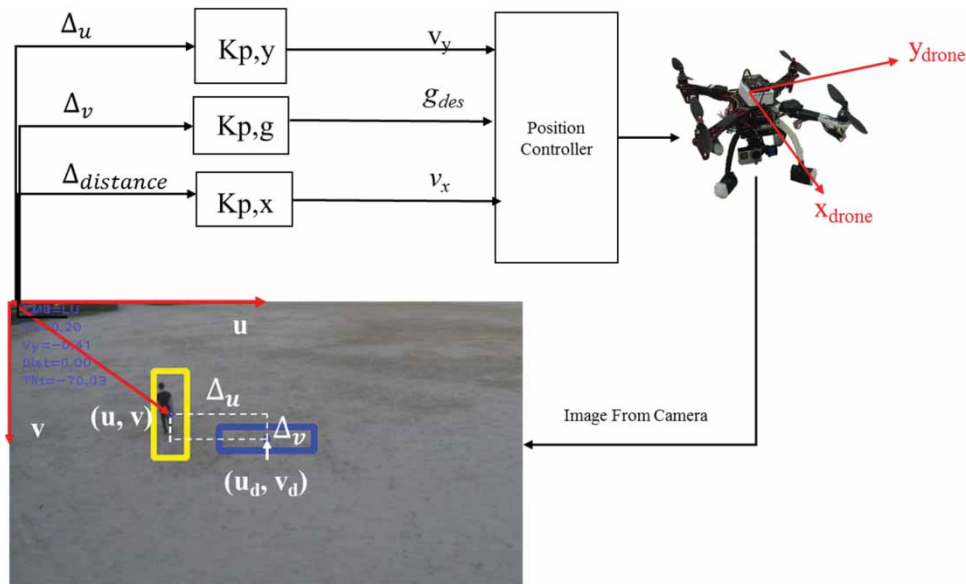
### 2.2.1. GPS tracking mode

Figure 5 shows the procedure of GPS tracking mode in GCS side. The GPS-based follow-me algorithm works based on the longitude, latitude and altitude values from the GCS. Together with the user moving direction and speed estimation which are provided by Android Location library, the GCS calculates the next position for the drone then commands the drone to move to GCS position (lon, lat, alt) plus the desired relative position vector  $\Delta(\text{lon}, \text{lat}, \text{alt})$ . Also, the gimbal tilt angle is computed based on the distance and altitude of the drone. The details of the control algorithm are described in Section 3.

### 2.2.2. Visual tracking mode

Our drone has five control parameters: (1)  $x$  and (2)  $y$  on drone's body frame coordinate, (3) altitude, (4) yaw (to the target) and (5) gimbal angle, in addition to 3 target variables (1)  $u$ , (2)  $v$ , and (3) size (3DoF) in the captured image. For simplicity of algorithm, we keep the altitude constant and yaw to face the front of the target, and use three control parameters, namely longitude, latitude, and gimbal angle, only. Figure 6 illustrates the mapping and control between pixel domain variables and errors in the physical control variable in the drone. We used simple linear feedback controllers for each of the three target variables that satisfies the current control purpose. The details of the control algorithm are discussed in Section 4. These three target variables are defined as follows:

- $v_y$  (horizontal): controls the drone's movement to left or right when user's bounding box deviates from the horizontal center of the frame



**Figure 6.** Drone control algorithm.

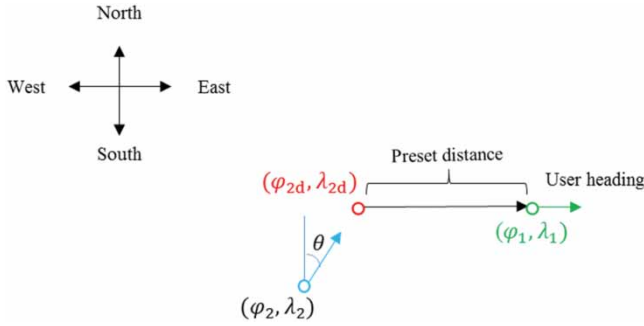
- $g$  (gimbal angle): to keep the user or target at the center of the image frame, it is adjusted when the center of user's bounding box moves vertically from the vertical center of the frame
- $v_x$  (*distance and target size*): controls the drone's movement in forward or backward direction when the drone is too far away or too nearer to the user. Theoretically, we can use the bounding box size to decide the object's distance but we applied a different approach explained in Section 4.

### 3. GPS-based tracking algorithm

In this section, we describe the conventional GPS-based follow-me algorithm and derive the mathematical model for calculating the visual positioning error from relative GPS errors.

#### 3.1. GPS-based follow-me operation

Figure 7 illustrates the mathematical model of the GPS-based follow-me operation. Let's denote  $\varphi_1, \lambda_1$  and



**Figure 7.** GPS-based follow-me operation.

$\varphi_2, \lambda_2$  latitudes and longitudes for GCS and drone position (in geographic coordinate system), respectively. First, we can get user heading from Android location provider. We denote user heading (bearing angle)  $\theta$ . Based on  $\theta$  and pre-set distance, we can calculate a destination point for the drone.

Because we want the drone to follow the user from behind, a destination bearing angle from user position to target point ( $\varphi_{2d}, \lambda_{2d}$ ) is different from user heading and can be calculated by:

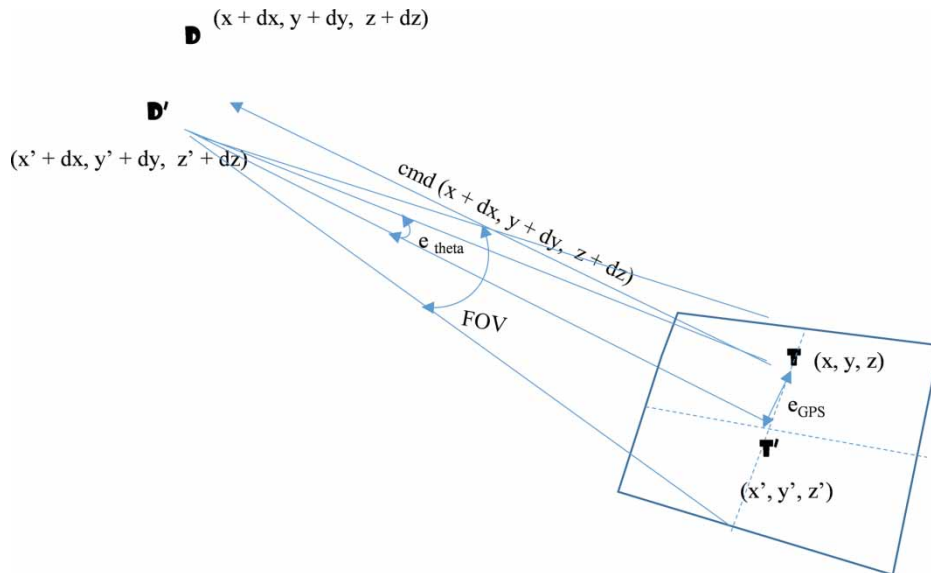
$$\theta_d = \begin{cases} \theta + 180 & \text{if } \theta < 180 \\ \theta - 180 & \text{if } \theta \geq 180 \end{cases} \quad (1)$$

The destination point for the drone is calculated based on the given distance and the destination bearing from the current user position:

$$\begin{aligned} \varphi_{2d} &= a \sin \left( \sin \varphi_1 \cos \frac{d}{R} + \cos \varphi_1 \sin \frac{d}{R} \cos \theta_d \right) \\ \lambda_{2d} &= \lambda_1 + a \tan 2 \left( \sin \theta_d \sin \frac{d}{R} \cos \varphi_1, \right. \\ &\quad \left. \cos \frac{d}{R} - \sin \varphi_1 \sin \varphi_{2d} \right). \end{aligned} \quad (2)$$

where:

- $\varphi_{2d}$  is destination latitude,
- $\lambda_{2d}$  is destination longitude,
- $d$  is pre-set distance
- $R$ : earth radius
- $\frac{d}{R}$  is the angular distance



**Figure 8.** GPS-based follow-me and camera geometry.



The destination point is then sent to the drone as its target position.

### 3.2. Accuracy model

Figure 8 illustrates the geometrical illustration of the drone and GCS. Coordinate system originates in the GCS center instead of longitude, latitude, and altitude. Since the absolute positioning value is not significant inline to our application scenario, without loss of generality, we can assume GCS GPS data has no error. Hence, the GPS difference errors are only due to the drone GPS error.

We use  $x, y, z$  for positions instead of latitude, longitude, and altitude. Let's denote  $T(x, y, z)$  as target user's true position,  $D(x + dx, y + dy, z + dz)$  as desired (from GCS) drone position,  $T'(x', y', z')$  as the estimated target user's position from the drone's GPS coordinate, and,  $D'(x' + dx, y' + dy, z' + dz)$  as the real drone position. The vector  $(dx, dy, dz)$  is the desired position of drone relative to the GCS. The camera's projection geometry maps the GPS difference error to the target camera angle error as follows:

$$\begin{aligned} e_{\text{theta}} &= \arcsin \left( \frac{|T - T'|}{|T - D'|} \right) \sim \arcsin \left( \frac{|T - T'|}{||T - D||} \right) \\ &= \arcsin \left( \frac{e_{\text{GPS}}}{\text{distance}} \right). \end{aligned} \quad (3)$$

Thus, the pixel errors in the captured video contents are calculated as:

$$\text{P.E. (pixel error)} \sim \left( \frac{\text{resolution}}{\text{FOV}} \right) \arcsin \left( \frac{e_{\text{GPS}}}{\text{distance}} \right). \quad (4)$$

### 3.3. GPS error experiment

To understand the degree of inaccuracy, we made preliminary GPS error measurement experiments with today's popular commercial GPS sensors, uBlox NEO-M8N of the drone and the GPS data of Smartphone (Samsung Galaxy Grand Max). Figure 9(a) shows the measured trajectories of two GPSs along the route (a stadium track) and Figure 9(b) shows the comparison of GPS different between commercial GPS sensors, uBlox NEO-M8N and GPS sensor of Samsung Galaxy Grand Max. The obtained results show the average distance of GPS is often over 2 m. GPS error also changes depending on the weather-condition, and environment and is found to be often larger than 5 m. The similar observations can be found in [4], where Hedgecock et al. reported 4–5 m error and developed a precise relative GPS estimation algorithm. We did not apply their algorithm because their

algorithm needs low-level sensor data from GPS sensors, which is not available in most of the commercial GPS sensors and also in Android or iOS platforms.

According to Equation (6), with the typical conditions of the camera with 60° FOV, 1k pixel (horizontal or vertical) resolution, and the target distance of 10 m, the P.E. is 200 pixels with 2 m GPSE (GPS error) and 500 pixels with 5 m GPSE. That is, with typical GPS errors, the target is captured more than 30% of time at out of the image center, however, error larger than 5 m the target is found to be out of camera frame. The experiment results in Section 5 show that video content of over 20% out-of-scope is usual.

## 4. Visual tracking algorithm

As described in Section 2, independent simple controllers are used for each of the three target variables, namely vertical, horizontal, and size control. Note that we focus on explaining the relation among pixel domain measurement, physical position and angle control rather than developing a robust control system.

### 4.1. Vertical (camera gimbal angle) control

The camera gimbal angle is controlled to place the target at the center of the captured image. To obtain the desired angle, we calculate the offset of the center of the bounding box from the center of image:

$$\Delta_v = b_v - f_v, \quad (5)$$

Where  $b_v$  is  $v$  (vertical) coordinate of the center of the current target's bounding box (in pixel), and  $f_v$  is  $v$  coordinate of image center point (in pixel).

Whilst the image size is fixed and known, to calculate the gimbal angle offset we need to know the ratio of pixel/angle of the camera. It means how many pixels the object moves vertically when the gimbal angle changes by 1°. The GoPro camera has a vertical field of view (V.FOV) of 37.2°. The ratio of degree/pixel  $\alpha$  can be calculated by:

$$\alpha = \frac{\text{V.FOV}}{\text{frame height}}. \quad (6)$$

The gimbal offset is calculated by:

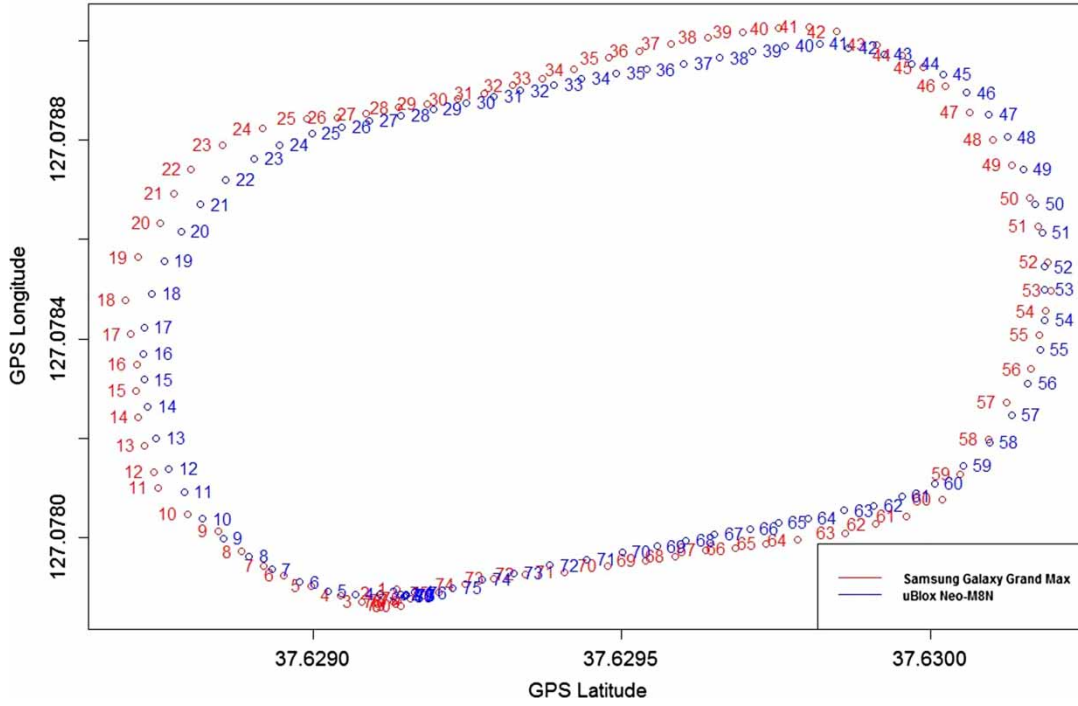
$$\Delta g = \Delta_v \alpha. \quad (7)$$

Destination angle value to control the gimbal:

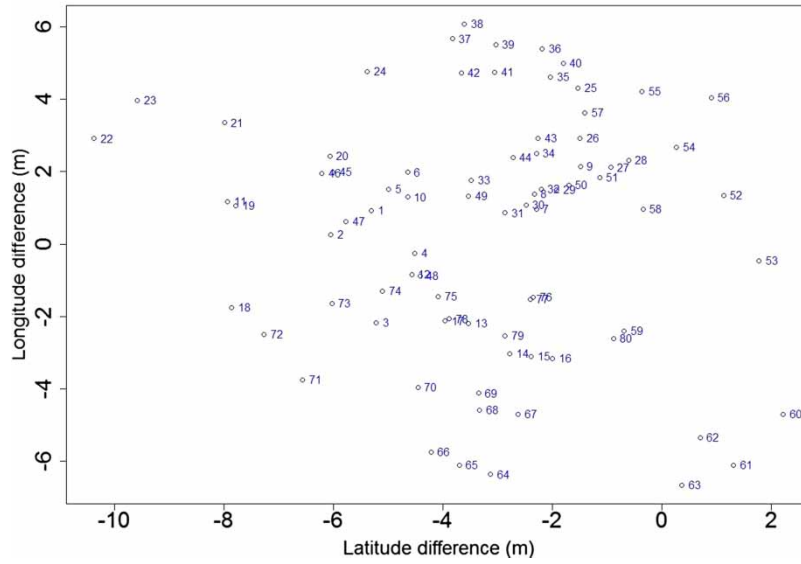
$$g_{\text{des}} = g + \Delta g. \quad (8)$$

In every frame, we repeat these steps to control the gimbal angle.

(a) Measured GPS data from an uBlox Neo-M8N and a Samsung Galaxy Grand Max along a stadium track



(b) GPS different between UBlox NEO-M8N and Samsung Galaxy Grand Max

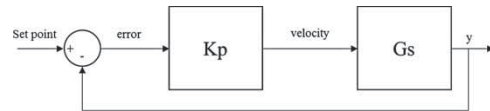
**Figure 9.** Measured trajectories of two GPS sensors along the same route: trajectories and the difference of GPS data.

#### 4.2. Horizontal control

The horizontal offset of the center of the bounding box from the image center is the measure of drone's location from a desired location relative to the target.

$$\Delta_u = b_u - f_u, \quad (9)$$

where  $b_u$  is  $u$  (horizontal) coordinate of the center of current target's bounding box (in pixel), and  $f_u$  is  $u$

**Figure 10.** Horizontal position control.

coordinate or image center point (in pixel). The offset from the center point is known. Here we apply the P controller to calculate the next velocity of drone (Figure 10).



**Figure 11.** Bounding box scale problem.

Velocity to control the drone is calculated by:

$$y' = Kp_y \Delta_u. \quad (10)$$

We use different algorithms for horizontal and vertical position control for two reasons. First, we only use the tilt (vertical angle) of the gimbal. Second, we control drone position instead of gimbal angle using horizontal target position.

#### 4.3. Zooming (Forward/backward control)

Ideally, we could use the size of the bounding box of the object to estimate the distance between the drone and object but we observed that the accuracy of bounding box size from TLD algorithm is not good enough. For example, the bounding box becomes even larger when the user is far away as in Figure 11. Therefore, we come up with another algorithm to estimate the distance from drone to user. Since GCS learns the current altitude and gimbal angle of the drone, the distance between the drone and target user can be calculated by Pythagoras's theorem as Equation (11).

$$d = h \tan(g). \quad (11)$$

Then, the algorithm for controlling in forward or backward direction is as follows:

- Step 1: At the beginning of follow-me mode, our algorithm controls the gimbal so that the user is inside of a desired area. The GCS calculates the distance between drone and user by Equation (11),  $d_{init} = h \tan(g)$
- Step 2: On every next frame, the GCS calculates the destination gimbal angle at the current frame, from that the GCS can calculate the current distance to the user by the equation,  $d_{current} = h \tan(g_{des})$
- Step 3: Compare current distance to initial distance,  $\Delta d = d_{current} - d_{init}$ .
- Step 4: From  $\Delta d$  and simple P controller like control left and right, the GCS can calculate the target velocity to control drone,  $x' = Kp_x \Delta_d$ , where  $x'$ : Velocity to control the drone move forward/backward
- Step 5: Send MAVLink control command to the drone to control it forward or backward.

The calculated  $x'$ ,  $y'$  and  $g_{des}$  are sent to drone by MAVLink protocol, then the drone controls its motors' speed accordingly.

## 5. Experimental results

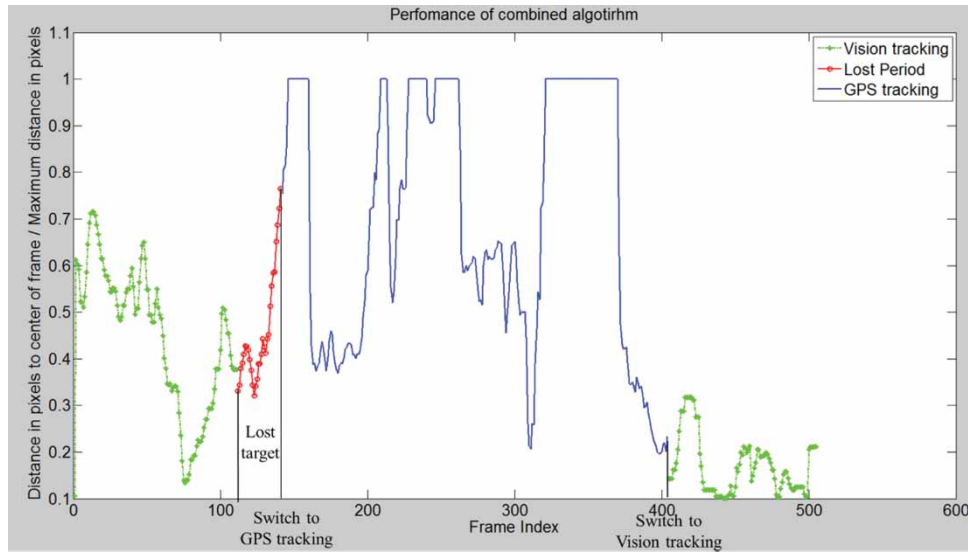
To demonstrate the validity of our design and implementation, scenario-based experiments are performed and discussed. First, two experiments verifying the system design and operation, and then the performance comparison results with GPS-based tracking are presented.

### 5.1. Switching between visual and GPS tracking mode

The scenario illustrated in Figure 12 tests the switching operations between visual and GPS tracking mode. The drone starts in visual tracking mode. Then the user on purpose hides behind the goal post so that visual tracker loses it. The user waits for the drone switch to GPS tracking mode. The user waits for the drone to redetect the



**Figure 12.** Switching mode test: (1) visual tracking mode, (2) the user hides after the goal post so that visual tracker loses it, (3) after 30 frames (our threshold: 6 s) the drone switches to GPS tracking mode. (4) The drone follows the user in GPS tracking mode. (5) When visual tracker redetects the target, the drone switches back to vision tracking mode.



**Figure 13.** Tracking mode switching test result: the distance of target center to the center of the visual frame.

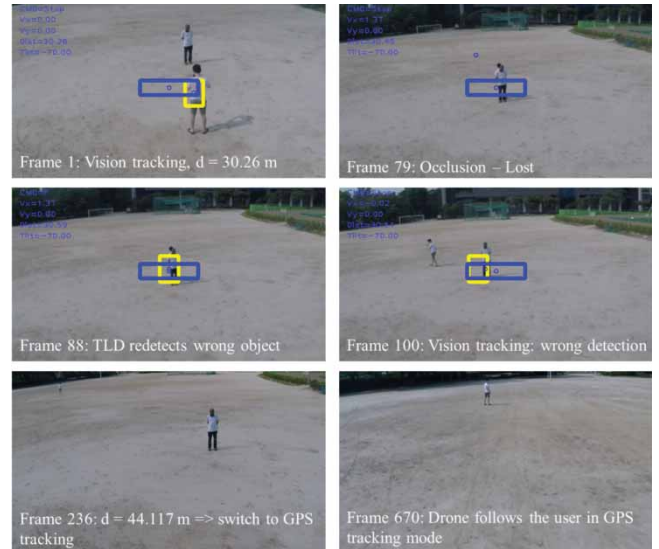
target. Figure 13 shows the key frames with augmented target bounding box.

Figure 13 shows the log results in the GCS system. From frame 1 to frame 111, the drone operated in visual tracking mode. The initial distance between drone and user is 9.82 m. At frame 112, the tracking algorithm loses the object due to object deformation. After 30 frames (6 s), at frame 142, the drone switches to GPS tracking mode. At frame 405, the tracking algorithm redetects the user, also the distance between the drone and the GCS is 10.78 smaller than the initial distance + threshold (5 m). Because both conditions satisfy, the drone switches back to the vision tracking mode.

## 5.2. Conflict between visual and GPS trackers

The scenario illustrated in Figure 14 tests when the visual tracker and GPS tracker conflicts. In the beginning, the system starts with visual tracking mode, the distance between the drone and the user is 30.26 m—calculated by the GPS data of the drone and the GCS. After the occlusion with another person, i.e. similar looking object, at the 79th frame, TLD visual tracker tracks the wrong object. The original target user moves further and the distance between the drone and the target user is 44.117 m. When the distance is larger than the initial distance (30.26 m) + 10 m (our algorithm's threshold), the tracker switches to GPS tracking algorithm. Figure 15 shows the distance between the drone and target is reduced continuously.

We included this experiment result to point out that our system provides reliability for unexpected operating conditions. When the drone recovers the distance to GCS and the view of the target user, almost always GCS



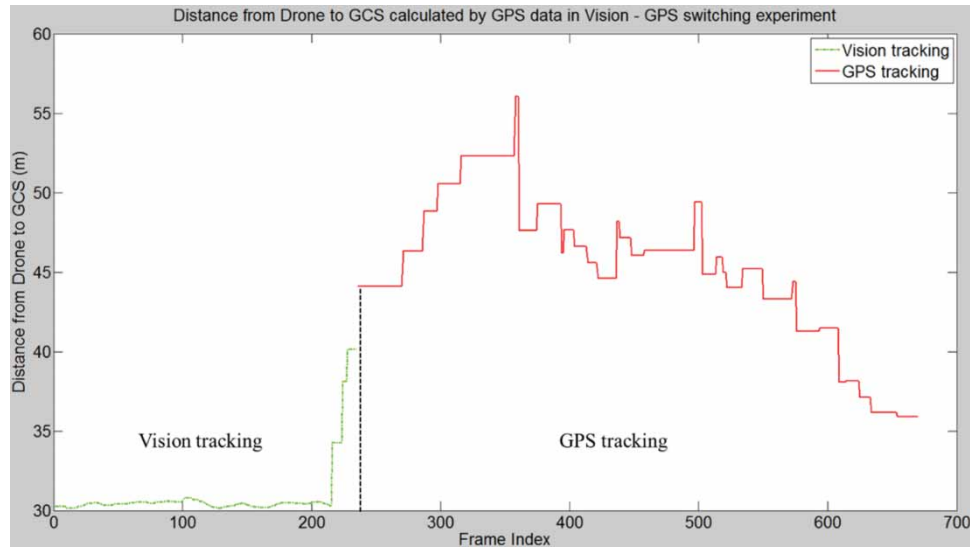
**Figure 14.** Vision-GPS switching caused by the wrong detection: (1) the tracker starts in visual tracking mode. (2) The target object is occluded by a similar object. (3) The object tracking algorithm then detects the similar wrong one. (4) The target object keeps moving. (5) The distance reached the threshold and the tracker switches to GPS tracking mode.

return to vision tracking mode, but once or two times we observed that GCS could not recover the Wi-Fi connection due to the unknown reasons. However, our system can maintain the tracking performance of GPS tracking mode.

## 5.3. Tracking accuracy

For important performance measure of follow-me operation, we compare target tracking accuracies between the GPS-based tracking and combined tracking algorithm.





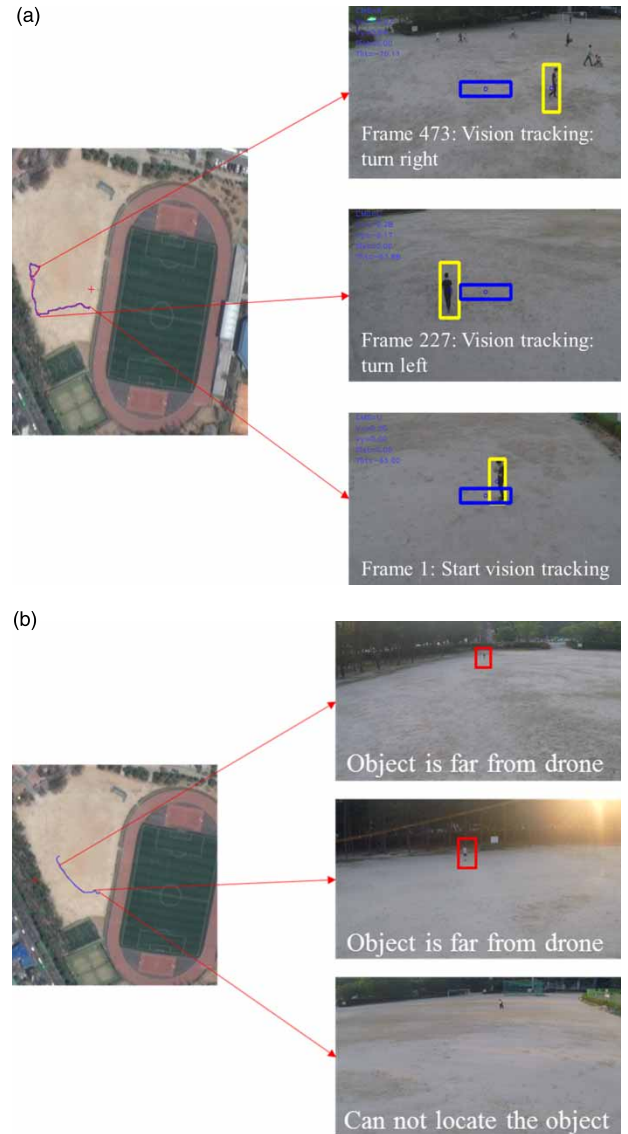
**Figure 15.** Conflict between visual and GPS trackers test result: distance of the target center to the center of the visual frame.

Only one typical experiment in Figure 16 is included in this paper, but several tests show a similar performance difference. Figure 16 shows the tracking path and snapshots in key positions for different modes. The reader could check the full video in the supplementary multimedia contents. As the snapshots show, the vision only tracking tracks the target much better in the position and size than GPS-only tracking till the target is lost. For comparison purpose, multiple tests with different following modes are performed in the same scenario and target path. Euclidean distance in pixel between object center and video frame center is used for performance criteria.

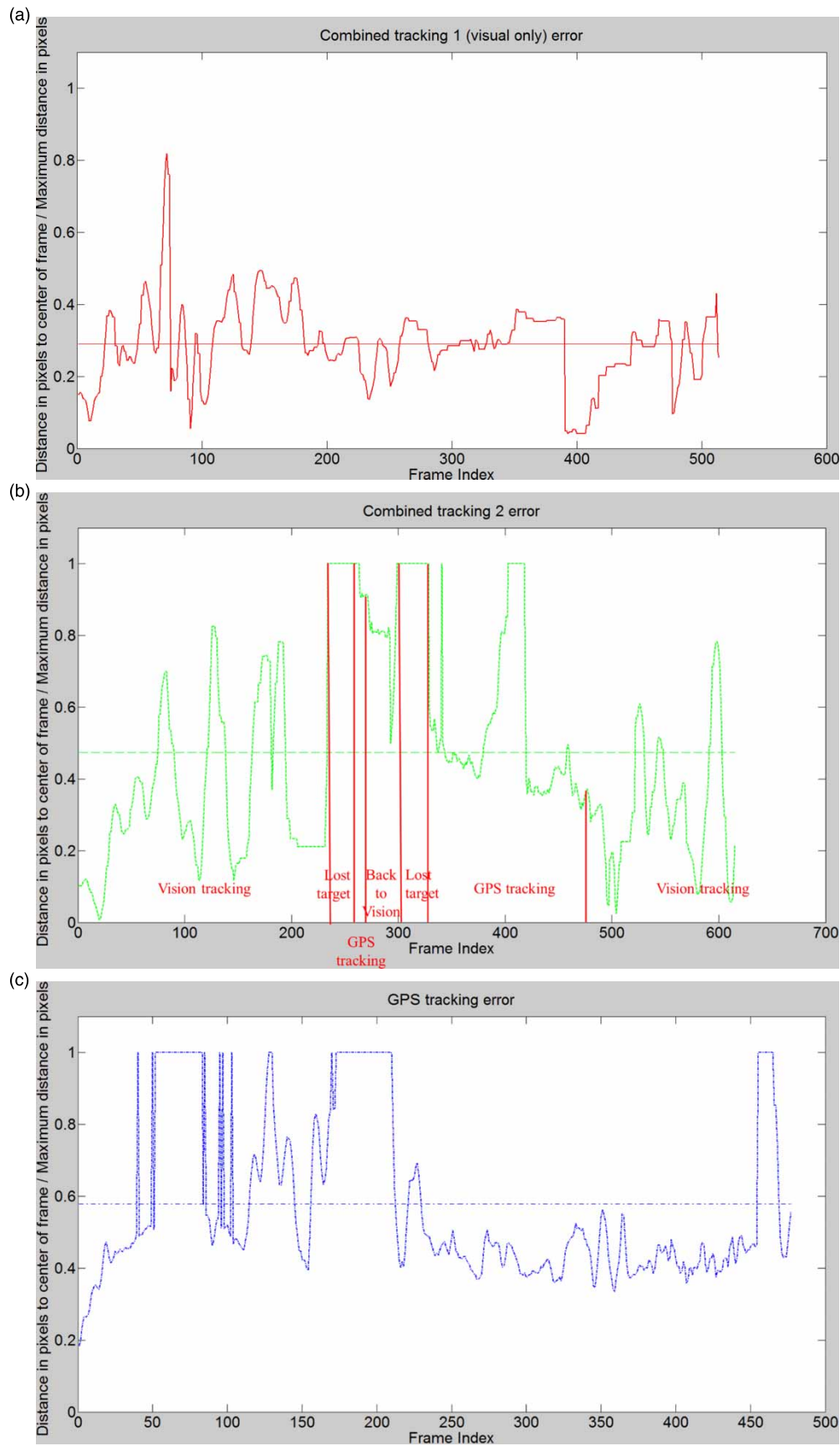
Figure 17 displays the ratio of the target position in the video frame to the maximum distance (i.e. from the center to the frame corner) from three test results, the solid line and the dash line stand for combined tracking and the dash-dot line depicts GPS tracking

In the first test of the combined algorithm (the solid line), the visual tracker did not lose the target, hence, all the data are from visual tracking. Let denote  $E$  is a distance in pixels to the center of frame/maximum distance in pixels and  $R$  is the number of out of frame/total number of the frame. In Combined 1 tracking experiment,  $E = 0.2898$  and  $R = 0$ .

In the second test of the combined algorithm (the dashed line), the visual tracking fails two times, the first time for Wi-Fi connection loss and the second time for tracking algorithm failure. As mentioned earlier, to reduce the computational load at GCS, one frame from every five video frames are processed. The horizontal axis is indexed in the vision processed frame, so five frames correspond to 1 s. In the beginning, the system starts with visual tracking mode and continues the mode till the 233rd frame when it loses the target due to Wi-Fi connectivity loss. The system tries to redetect the target in



**Figure 16.** ROI tracking data: (a) visual tracking, (b) GPS tracking.



**Figure 17.** Comparison of the tracking accuracy between GPS only and combined. (a) Combined tracking 1 (visual only), (b) Combined tracking 2, (c) GPS only tracking.



the next 30 frames till the 274th frame, but failed and switches to GPS tracking mode. The system switches back to visual tracking mode at frame 283rd when it re-detects the target. At 303rd frame, it loses the target one more time due to visual tracking failure, it continues following the user in GPS mode from 333rd to 489th frame. After that, it finds the target again, switches back to vision tracking algorithm and follows the target. In this experiment,  $E = 0.4742$  and  $R = 77/615 = 12.5\%$

GPS tracking experiment result shows that the target sometimes goes out of the video frame.  $E = 0.5790$  and  $R = 91/477 = 19.07\%$  in this case.

The data show significant performance improvement: (1) up to 50% error reduction during visual tracking mode and (2) 20% more time the target object resides in the center area of the frame. The target was kept inside the frame for the whole experiment during visual tracking mode. Meanwhile, the target often goes out of the video frame in the GPS tracking algorithm. Note we use distance value 1.0 for out-of-view cases. The supplementary video contents are included to show the video contents of GPS-based and combined (with failures).

## 6. Conclusion

In this paper, we presented a Visual-GPS combined tracking algorithm for follow-me application of selfie drones and demonstrated its superior performance and robustness when compared to the existing GPS-based solutions. Our algorithm combines the respective strength of GPS and vision tracking that helps the drone in tracking and following the target accurately in long-term. As shown in the experiments, the tracking performance of the GPS-based algorithm does not reach the eye-level of users at the moment. Often over 20% of times it loses the target. The visual tracking can localize the target very accurately once the target enters into the camera FOV and the algorithm detects it, whereas it completely loses the location when it loses the target.

Despite demonstration of the usefulness of our approach, the current implemented prototype has several limitations for commercial use. First, the current system provides many sub-modes of Follow-me operations, such as leash, lead, left, right, and circle mode. But all the modes can be extended easily based on the leash mode. Additionally, sometimes the Wi-Fi connection loss happens due to the limitation of GCS antenna performance. This weakness can be mitigated with specially designed long-range Wi-Fi in the GCS system.

## Disclosure statement

No potential conflict of interest was reported by the authors.

## Funding

This work was funded by the Ministry of Trade, Industry and Energy, Korea [Project no: 10050063, Project Name: Development of drone system with a built in high resolution video and multi-channel recorder for detection actively mover].

## Notes on contributors

**T. Tuan Do** is currently a Software Engineer at YouVR, Korea. He received his BS degree in 2013 from HUST (Hanoi University of Science and Technology), Vietnam and MS degree in 2017 from SeoulTech (Seoul National University of Science and Technology). Before attending SeoulTech in 2015, he worked as a software developer at Samsung Mobile Center, Hanoi, Vietnam. He published two Korean journal publications. His research interests include mobile software and image processing.

**Heejune Ahn** is currently a Professor in Department of Electrical and Information Engineering, SeoulTech (Seoul National University of Science and Technology). He received his BS, MS, and PhD degrees all from KAIST (Korea Advanced Institute of Science and Technology) in 1993, 1995, and 2000, respectively. Before joining SeoulTech in 2004, he worked as a wireless protocol engineer at LG Electronics, Korea, and as a software architecture at TmaxSoft, Korea. He published over 20 international journal publications and 1 US patent and many Korean journal publications and patents. His research interests include computer vision, visual communications, and software architecture design.

## ORCID

Heejune Ahn  <http://orcid.org/0000-0003-1271-9998>

## References

- [1] Schneider D. Flying selfie bots. *IEEE Spectr.* **2015**;52(1): 49–51.
- [2] Lim H, Park J, Lee D, et al. Build your own quadrotor: open-source projects on unmanned aerial vehicles. *IEEE Robot Autom Mag.* **2012**;19(3):33–45.
- [3] Mahony R, Kumar V, Corke P. Multirotor aerial vehicles. *IEEE Robot Autom Mag.* **2012**;19(3):20–32.
- [4] Hedgecock W, Maroti M, Sallai J, et al. High-accuracy differential tracking of low-cost GPS receivers. *The 11th ACM annual international conference on mobile systems, applications, and services*; Taipei, Taiwan; **2013**. p. 221–234.
- [5] Wu Y, Lim J, Yang M-H. Online object tracking: a benchmark. *The IEEE conference on computer vision and pattern recognition*; Portland, Oregon; **2013**. p. 2411–2418.
- [6] Wu Y, Lim J, Yang MH. Object tracking benchmark. *IEEE T Pattern Anal.* **2015**;37(9):1834–1848.
- [7] Kristan M, Matas J, Leonardis A, et al. The visual object tracking vot2015 challenge results. *The IEEE international conference on computer vision workshops*; Boston, Massachusetts; **2015**. p. 1–23.

- [8] Kalal Zdenek, Mikolajczyk K, Matas J. Tracking-learning-detection. *IEEE T Pattern Anal.* **2012**;34(7):1409–1422.
- [9] Hare S, Golodetz S, Saffari A, et al. Struck: structured output tracking with kernels. *IEEE T Pattern Anal.* **2016**;38(10):2096–2109.
- [10] Zhong W, Lu H, Yang MH. Robust object tracking via sparsity-based collaborative model. *IEEE Conference on Computer vision and pattern recognition (CVPR)*; Providence, Rhode Island, USA; **2012**. p. 1838–1845.
- [11] Pestana J, Sanchez-Lopez J, Campoy P, et al. Vision based GPS-denied object tracking and following for unmanned aerial vehicles. **2013 IEEE International Symposium on Safety, Security, and Rescue Robotics (SSRR)**; Linköping, Sweden; **2013**. p. 1–6.
- [12] Bart R, Vykovsk A. Any object tracking and following by a flying drone. **2015 Fourteenth Mexican International Conference on Artificial Intelligence**; Cuernavaca, Mexico; **2015**. p. 56–59.
- [13] DJI Machine Learning lab. Phantom 4 active track algorithm test and comparison: sports car online. **2016 Jun 1**. Available from: <http://www.youtube.com/watch?v=Vja83KLQXZs>.
- [14] Conrad Toles. Mavic pro active track fail!; **2017 Apr 12**. Available from <https://www.youtube.com/watch?v=dhHaVE4wvFc>.
- [15] Meier L, Camacho J, Godbolt B, et al. Mavlink: micro air vehicle communication protocol. Available from: <http://qgroundcontrol.org/mavlink/start>.