Article

Drone State Estimation Based on Frame-to-Frame Template Matching with Optimal Windows

[Seokwon Yeom](#) *

Article

Drone State Estimation Based on Frame-to-Frame Template Matching with Optimal Windows

Seokwon Yeom

Dept. of Artificial Intelligence, Daegu University, Gyeongbuk, South Korea; yeom@daegu.ac.kr

Abstract: The flight capability of drones expands the surveillance area and allows drones to be mobile platforms. Therefore, it is important to estimate the kinematic state of the drone. In this paper, the kinematic state of a mini drone in flight is estimated based on the video captured by the camera. The instantaneous velocity of the drone is measured through image-to-position conversion and frame-to-frame template matching using optimal windows. Multiple templatpes are defined by their corresponding windows in a frame. The size and position of the windows are obtained by minimizing the sum of the least square errors between the piecewise linear regression model and the nonlinear image-to-position conversion function. The displacement between two consecutive frames is obtained via frame-to-frame template matching that minimizes the sum of normalized squared differences. The kinematic state of the drone is estimated by a Kalman filter based on the measured velocity. The Kalman filter is augmented to simultaneously estimate the state and velocity bias of the drone. For faster processing, a zero-order hold scheme is adopted to reuse the measurement. In the experiments, two 150-meter-long roadways were tested; one road is in an urban environment and the other in a suburban environment. A mini drone starts from a hovering state, reaches top speed, and then continues to fly at a nearly constant speed. The drone captures video 10 times on each road from a height of 40 m at a 60-degree camera tilt angle. It will be shown that the proposed method achieves average distance errors at low meter levels after the flight.

Keywords: drone state estimation; image-position conversion; frame template matching; optimal windows; Kalman filter; bias estimation

1. Introduction

Drones can hover or fly while capturing videos from a distance. They can cover large and remote areas. This observation can be made from various viewpoints and altitudes [1–3]. They are also cost-effective and do not require highly trained personnel [4]. Flying drones also act as mobile sensing platforms; drones collect data during flight equipped with various sensors [5,6]. Therefore, the position and velocity information of drones are essential for navigation, surveillance, and other high-level tasks. For example, drones can follow a desired trajectory, track multiple targets, and optimize formation in swarm operations.

The drone's position is usually estimated by external sensors such as GPS and internal sensors such as IMU. GPS provides absolute position information but is vulnerable to external conditions [7,8]. IMU provides fast updates and require no external infrastructure, but its estimation performance degrades quickly due to drift errors that accumulate over time [8]. Localization methods using LiDAR or depth cameras have been developed, but they are typically limited to low-altitude or indoor conditions [9–12]. The RF signal sensor-based UAV navigation often requires pre-installed transmitters for accurate localization [13].

Vision-based localization techniques often rely on object template or feature matching through images [14]. Frame-to-frame template matching is commonly used to estimate displacement between consecutive frames [15]. It can track motion by matching a template from the previous frame to the current frame. Standard template matching works well for small displacements in

environments without lighting, rotation, or scale changes [16]. A template is a region of interest (ROI) selected from a fixed area or object in the first frame or a reference image. Template selection may depend on terrain, objects, or explicit features [14,17]. Moving foreground regions were used as automatic template candidates with gaussian mixture modeling of the background [18]. Photometric property-based template selection was developed to choose templates depending on the intensity, contrast, or gradient of pixels [17,19]. Selecting a proper template is critical for accurate motion estimation. As scenes or objects change over time, templates need to be updated however, incorrect updates can lead to accumulated errors that degrade performance [20].

In the paper, the kinematic state of a drone in flight is estimated only from frames captured by the drone. The frame-to-frame template matching using optimal windows is proposed to measure the instantaneous velocity of the drone. The optimal window divides the frame into several regions to minimize the non-uniform spacing of the real coordinates.

Imaging projects 3D space on a 2D plane, and this projection can be modeled using principles of ray optics [21]. The image-to-position conversion converts the integer coordinates of pixels into continuous real-world coordinates [22]. During the conversion, the image size, the camera's horizontal and vertical angular field of view (AFOV), elevation, and tilt angle are assumed to be known. However, this conversion process generates non-uniform spacing in real-world coordinates. When the camera points straight down, the pixel spacing is the most uniform. In [23], an entire frame is set as a template to estimate the drone's speed from the vertical view. However, the spatial and visual information is one-sided, and the surveillance area becomes narrower in the vertical view. The optimal windows are contrived to overcome the non-uniform spacing distortion in the real-world coordinates. The height and location of the optimal windows are obtained from the piecewise linear segments that best fit the image-to-position conversion function in the vertical direction. The split points of the segments are determined so as to minimize the sum of least square errors of the seperate linear regression lines [24,25]. Therefore, multiple templates are independent of the scene and object. No additional process is required for the template update. The matching of each template is performed by minimizing the sum of normalized squared differences [26], and the instantaneous velocity is calculated from the average displacement obtained through multiple template matching.

The drone's kinematic state is estimated based on the measured velocity using a Kalman filter, which adopts a nearly constant acceleration (NCA) model [27]. The state of the Kalman filter is augmented to simultaneously estimate the drone's state and bias in velocity [28]. Since the computational complexity of frame matching is high, the augmented-state Kalman filter with a zero-order hold scheme [29] on the measurements is adopted for faster processing. The zero-order hold Kalman filter reuses the measurement until new measurements are available.

Figure 1 shows a block diagram of the proposed method. First, the drone's velocity is measured by image-to-position conversion and frame-to-frame template matching using optimal windows. The measured instantaneous velocity is input to the Kalman filter. Next, the drone's state and bias in the velocity are estimated through the augmented Kalman filter.

Figure 1. Block diagram of drone state estimation.

In the experiment, a mini drone weighing less than 250 g [30] flies along two approximately 150 m long roads and captures video at 30 frame per second (FPS). One road is an urban environment, the other is a suburban environment. The drone starts from a stationary hovering position, accelerates to maximum speed, and continues to fly at a nearly constant speed at an altitude of 40 m and a camera tilt angle of 60 degrees. The frame size is 3840x2160 pixels, and the AFOV of the camera is set to 64 and 40 degrees in the horizontal and vertical directions, respectively. Ten flights were repeated on each road under various traffic conditions. For faster processing, three additional frame matching

speeds (10, 3, 1 FPS) are tested using the zero-order hold Kalman filter. The proposed method is shown to achieve an average flight distance errors of 3.07-3.57 m and 1.97-2.39 m for Roads 1 and 2, respectively.

The contributions of this study are as follows: (1) Multiple template selection using optimal windows is proposed. The optimal windows are determined only by the image size, the camera's AFOV, elevation, and tilt angle. Therefore, the template selection process is independent of the scene or object. (2) The augmented-state Kalman filter is designed to improve the accuracy of the drone's state. The drone's flight distance is estimated with high accuracy, resulting in low-meter-level average errors, (3) Real-time processing is possible with the zero-order hold Kalman filter. This method maintains similar error levels even when the frame matching speed is reduced to 1 FPS.

The rest of the paper is organized as follows: the real-world conversion and frame-to-frame template matching with optimal windows are described in Section 2. Section 3 explains drone state estimation with the augmented state Kalman filter. Section 4 presents the experimental results. Discussion and conclusions follow in Sections 5 and 6, respectively.

2. Drone Velocity Measurement

This section describes how the instantaneous velocity is measured by frame-to-frame template matching with optimal windows.

2.1. Image-to-Position Conversion

The image-to-position conversion [22] applies trigonometry to compute real-world coordinates from image pixel coordinates assuming that the camera's AFOV, elevation, and tilt angles are known and the camera rotates only around the pitch axis. It provides a simple and direct conversion from pixel coordinates to real-world coordinates. However, the non-uniform spacing in the real-world coordinates arises as the tilt angle is larger and the altitude lower. The real-world position vector $\boldsymbol{x}_{ij}$ corresponding to the (i, j) pixel is calculated as [22],

$$x_{ij} = (x_i, y_j) \approx \left(d_{\frac{H}{2}} \cdot \tan\left[\left(i - \frac{W}{2} + 1\right)\frac{a_x}{W}\right], \quad h \cdot \tan\left[\theta_T + \left(\frac{H}{2} - j\right)\frac{a_y}{H}\right]\right),$$

$$i = 0, \dots, W - 1, \ j = 0, \dots, H - 1, \tag{1}$$

where W and H are the image sizes in the horizontal and vertical directions, respectively, h is the altitude of the drone or the elevation of the camera, a_x and a_y are the camera AFOV in the horizontal and vertical directions, respectively, and θ_T is the tilt angle of the camera. $d_{H/2}$ is the distance from the camera to $(x_{W/2-1}, \ y_{H/2}, 0)$, which is $\sqrt{y_{H/2}^2 + h^2}$. Figure 2 illustrates the coordinate conversion from image to real-world [22].
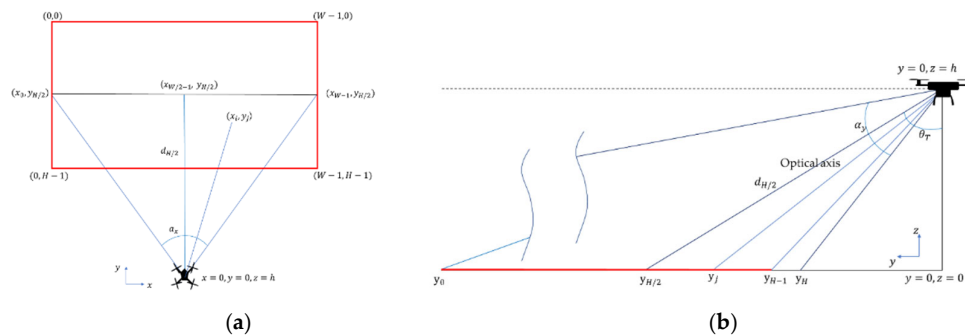

(a) (b)

Figure 2. Illustrations of coordinate conversion from image to real-world, (**a**) horizontal direction; (**b**) vertical direction.

Figure 3 visualizes the coordinate conversion function in horizontal and vertical directions according to Equation (1): W and H are set to 3840 and 2160 pixels, respectively; a_x and a_y are set

to 64° and 40°, respectively; h is set to 40 m; θ_T is set to 60°. The nonlinearity increases rapidly as pixels move away from the center, resulting in non-uniform spacing in real-world coordinates, especially in Figure 3(b). This distortion should be remedied when calculating the actual displacement in the image. In the next subsection, we will see how to overcome these distortions using optimal windows.
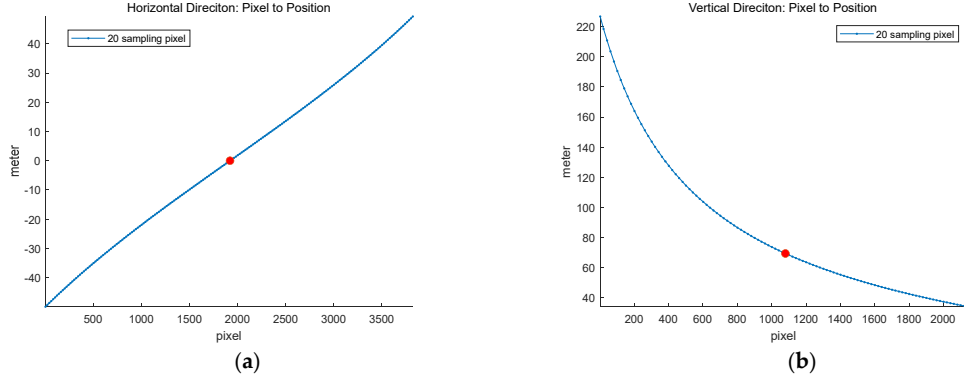

Figure 3. Coordinate conversion functions: (**a**) horizontal direction; (**b**) vertical direction. The red circle indicates the center of the image.

2.2. Frame-to-Frame Template Matching with Optimal Windows

The instantaneous velocity is measured at k frame as

$$\mathbf{z}_m(k) = \begin{bmatrix} z_{mx}(k) \\ z_{my}(k) \end{bmatrix} = \frac{1}{T}\frac{1}{N_w} \sum_{n=1}^{N_w} [\mathbf{c}_n - \mathbf{p}_n(k)], \tag{2}$$

$$\mathbf{c}_n = img2pos\left\{\begin{bmatrix} c_{nx} \\ c_{ny} \end{bmatrix}\right\}, \tag{3}$$

$$\mathbf{p}_n(k) = img2pos\left\{\begin{bmatrix} p_{nx}(k) + \dfrac{W}{2} \\ p_{ny}(k) + \dfrac{H}{2} \end{bmatrix}\right\}, \tag{4}$$

where T is the sampling time between two consecutive frames, N_w is the number of optimal windows, or equivalently, the number of templates, '*img2pos*' denotes the conversion process as in equation (1), and (c_{nx}, c_{ny}) is the center of the n-th window in pixel coordinates. In the experiments, c_{nx} is set to $W/2$, and c_{ny} is set to the center of the n-th linear segment. $(p_x\ p_y)$ is the displacement vector in pixel coordinates that minimizes the normalized sum of squared differences as

$$\begin{bmatrix} p_{nx}(k) \\ p_{ny}(k) \end{bmatrix} = argmin_{x,y}\left[\frac{\sum_{x',y'\in W_n}[I(x+x', y+y';k) - TMP_n(x', y';k-1)]^2}{\sqrt{\{\sum_{x',y'\in W_n} I(x+x', y+y';k)^2 \sum_{x',y'\in W_n} TMP_n(x', y';k-1)^2\}}} \right], \tag{5}$$

$$TMP_n(x,y;k) = I(x,y;k), \quad if\ x,y \in W_n, \tag{6}$$

where W_n indicates the n-th window, and I is the gray-scaled frame.

The image-to-position conversion function in the vertical direction is approximated by piecewise linear segments. The vertical length of each window is equal to the interval of each segment. The split points between segments are determined so that the sum of the least square errors of the separate linear models is minimized as follows:

$$\hat{s}_1, \ldots, \hat{s}_{N_w-1} = argmin_{s_1,\ldots,s_{N_w-1}} \left[\sum_{n=0}^{N_w-1} \sum_{j=s_n}^{s_{n+1}-1} min_{a_n,b_n} \left[y_j - (a_n j + b_n) \right]^2 \right], \tag{7}$$

where $s_1, \ldots, s_{N_w-1}$ are $N_w - 1$ split points for N_w windows, and s_0 and s_{N_w} are equal to 0 and H, respectively, and a_n and b_n are the coefficients of the n-th linear regression line. The number of windows is important. If there are too many windows, the sampling points (pixels) of one window may be too small, resulting in inaccurate displacements. If there are too few windows, the uneven spacing cannot be compensated for.

In the experiments, the number of windows is predetermined by 3 in the upper part and 2 in the lower part as it is desirable for the windows to be large and similar in size. The frame was cropped by 180 pixels near the edges to remove distortions that might occur during capture, resulting in optimal windows in an area of 3480 x 1800 pixels. Figure 4(a) shows the three linear regression lines of the vertical conversion function in the upper part, and Figure 4(b) shows the two linear regression lines of the lower part. Figure 5 shows 5 optimal windows and 4 split points on ae sample frame.
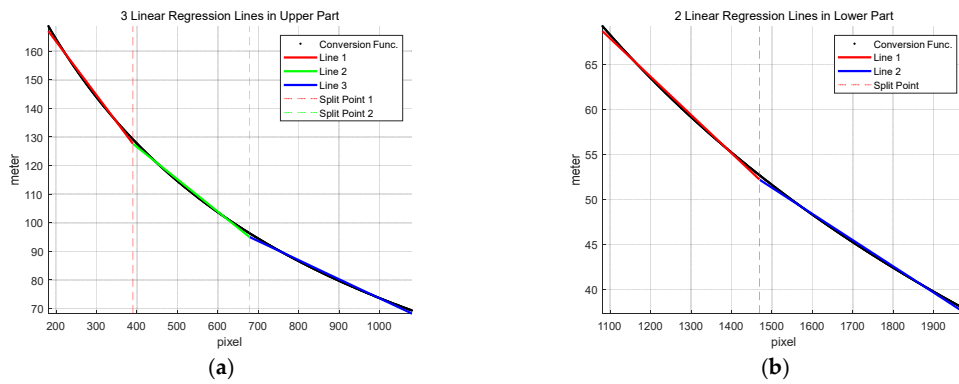

Figure 4. (a) three linear regression lines and two split point in the upper part, (b) two linear regression lines and one split point in the lower part.

Figure 5. Five optimal windows in the sample frame.

3. Drone State Estimation

3.1. System Modeling

The following augmented-state NCA model is adopted for the discrete state equation of a drone:

$$x(k + 1) = F(T)x(k) + q(T)v(k) + q_n n(k), \tag{8}$$

$$F(T) = \begin{bmatrix} 1 & T & \frac{T^2}{2} & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & T & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & T & \frac{T^2}{2} & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & T & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}, \quad q_v(T) = \begin{bmatrix} \frac{T^2}{2} & 0 \\ T & 0 \\ 0 & 0 \\ 0 & \frac{T^2}{2} \\ 0 & T \\ 0 & 0 \end{bmatrix}, \quad q_n = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 1 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 1 \end{bmatrix}, \quad (9)$$

where $x(k) = [x(k)\ \dot{x}(k)\ \ddot{x}(k)\ b_x(k)\ y(k)\ \dot{y}(k)\ \ddot{y}(k)\ b_y(k)]^t$ is the state vector of the drone at frame k, $x(k)$ and $y(k)$ are positions in the x and y directions, respectively, $\dot{x}(k)$ and $\dot{y}(k)$ are velocities in the x and y directions, respectively, $\ddot{x}(k)$ and $\ddot{y}(k)$ are accelerations in the x and y directions, respectively, and $b_x(k)$ and $b_y(k)$ are biases in the x and y directions, respectively. $v(k) = [v_x(k)\ v_y(k)]^t$ is a process noise vector, which is Gaussian white noise with the covariance matrix $Q_v = diag([\sigma_{vx}^2\ \sigma_{vy}^2])$, and $n(k) = [n_x(k)\ n_y(k)]^T$ is a bias noise vector, which is Gaussian white noise with the covariance matrix $Q_n = diag([\sigma_{nx}^2\ \sigma_{ny}^2])$. The measurement equation is as follows

$$z(k) = \begin{bmatrix} z_x(k) \\ z_y(k) \end{bmatrix} = Hx(k) + w(k), \quad (10)$$

$$H = \begin{bmatrix} 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 \end{bmatrix}, \quad (11)$$

where $w(k)$ is a measurement noise vector, which is Gaussian white noise with the covariance matrix $R = diag([r_x^2\ r_y^2])$.

3.2. Kalman Filtering

The state vector and covariance matrix are initialized, respectively, as follows

$$\hat{x}(0|0) = \begin{bmatrix} 0 \\ z_{mx}(0) \\ b_x(0) \\ 0 \\ z_{my}(0) \\ b_y(0) \end{bmatrix}, \quad P(0|0) = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & P_x & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & P_y \end{bmatrix}, \quad (12)$$

where $z_{mx}(0)$ and $z_{my}(0)$ are the measurements obtained from equation (2), $b_x(0)$ and $b_y(0)$ are the initial biases in x and y directions, respectively, and P_x and P_x are the initial covariances of the bias in x and y directions, respectively. The state and covariance predictions are iteratively computed as

$$\hat{x}(k|k-1) = F\hat{x}(k-1|k-1), \quad (13)$$

$$P(k|k-1) = FP(k-1|k-1)F^t + q_v(T)Q_v q_v(T)^t + q_n Q_n q_n^t. \quad (14)$$

Then, the state and covariance are updated as

$$\hat{x}(k|k) = \hat{x}(k|k-1) + W(k)[z_m(k) - H\hat{x}(k|k-1)], \quad (15)$$

$$P(k|k) = P(k|k-1) - W(k)S(k)W(k)^T, \quad (16)$$

where the residual covariance $S(k)$ and the filter gain $W(k)$ are obtained as

$$S(k) = HP(k|k-1)H^t + R, \quad (17)$$

$$W(k) = P(k|k-1)H^t S(k)^{-1}. \quad (18)$$

When the zero-order hold scheme is applied, the measurement $z_m(k)$ in equation (15) is replaced by $z_{zoh}(k)$ as

$$
\mathbf{z}_{zoh}(k) = \begin{cases} \mathbf{z}_m(1), & 1 \le k < L-1, \\ \mathbf{z}_m(L), & L \le k < 2L-1, \\ \quad \vdots \\ \quad \vdots \end{cases},
\tag{19}
$$

where *L*-1 is the number of frames before a new frame matching occurs, thus the frame matching speed is frame rate (frame capture speed) divided by *L*.

4. Results

4.1. Scenario Description

A mini drone weighs less than 250 g (DJI Mini 4K) [30] flies along two different 150-meter long roads and captures videos at 30 FPS with a frame size of 3840x2160 pixels. Figure 6(a) shows a commercially available satellite image of a vehicle road in an urban environment (Road 1) while Figure 6(b) shows a vehicle road in a suburban environment (Road 2). The urban road has more complex backgrounds than the suburban road. The drone starts from a stationary hovering point O, reaches its maximum speed in normal flight mode before Point A, and then continues flying at a nearly constant speed passing near Points A, B, and C. For Road 1 and Road 2, the distances between Point O and Point A are approximately 57 m and 48 m, respectively, the distances between Point O and Point B are approximately 109 m and 100 m, respectively, and the distances between Point O and Point C are approximately 159 m and 150 m, respectively. The flights were repeated 10 times in different traffic conditions on each road. The drone altitude was set to 40 m, and the camera tilt angle was set to 60 degrees. The AFOV was assumed to be 64 degrees and 40 degrees in the horizontal and vertical directions, respectively.
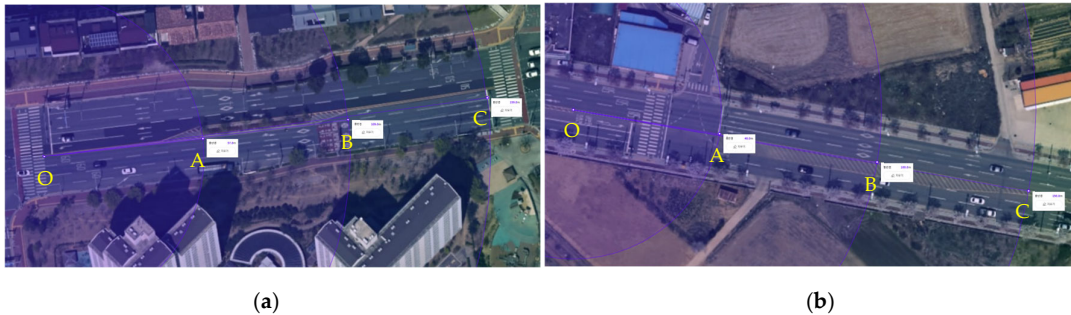

(**a**)   (**b**)

Figure 6. Satellite image of (**a**) Road 1, (**b**) Road 2.

Figure 7(a) and 7(b) show sample video frames when the drone passes near Points O, A, B, and C on Road 1 and 2, respectively. The optical center of the camera is marked with a white '+', which is assumed to be the drone's position.

(**a**)

(**b**)

Figure 7. Sample frames when the drone passing near Points O, A, B, C on (**a**) Road 1, (**b**) Road 2.

4.2. Drone State Estimation

Table 1 shows the parameter values of the augmented-state Kalman filter. The sampling time is set to 1/30 s. the process noise standard deviation is set to 5 m/s² in both x and y directions, the measurement noise standard deviation is set to 3 m/s in both x and y directions, and the bias noise standard deviation is set to 0.01 and 0.1 m/s in x and y directions, respectively. The initial covariance is set to the identical matrix except for the bias factors, which are set to 0.1 m²/s² in both x and y directions. The initial bias in the x direction is set to 0 m/s for both roads while the initial bias in the y direction is set to -1.2 to 0.2 m/s depending on the road traffic conditions and the frame matching speed. The 10 videos of Road 2 are divided into three groups and different initial values are applied as shown in Table 2.

Table 1. Model Parameters.

Parameter (Unit)	Road 1	Road 2
Sampling Time (T) (second)	1/30	
Process Noise Std. (σ_{vx}, σ_{vy}) (m/s^2)	(3, 3)	
Bias Noise Std. (σ_{nx}, σ_{ny}) (m/s)	(0.01, 0.1)	
Measurement Noise Std. (r_x, r_y) (m/s)	(2, 2)	
Initial Bias in x direction ($b_x(0)$) (m/s)	0	
Initial Covariance for Bias (P_x, P_y) (m^2/s^2)	(0.1, 0.1)	

Table 2. Initial Bias in y direction ($b_y(0)$) (m/s).

Frame Matching Speed (FPS)	Road 1	Road 2		
		Group 1	Group 2	Group 3
30				
10	-0.7	-0.3	0	0.2
3				
1	-0.8	-0.7	-0.3	-0.1

Tables 3 and 4 show the distance errors from Point O to Point C of 10 videos for Roads 1 and 2, respectively. Distance errors are computed using measured or estimated velocities at various frame matching speeds from 30 to 1 FPS. The average distance error based on the measured velocity is 11.99 m to 17.20 m for Road 1 and 4.95 m to 5.79 m for Road 2. The average distance error based on the estimated velocity for Road 1 is 3.07 m to 3.52 m and the average distance error for Road 2 is 1.97 m to 2.39 m. As the frame matching speed decreases, the average distance errors of the measured velocities increase, but the average distance errors of the estimated velocities remain at a similar level showing the robustness of the proposed system.

Table 3. Distance Errors to Point C on Road 1.

FPS	Type	Video 1	Video 2	Video 3	Video 4	Video 5	Video 6	Video 7	Video 8	Video 9	Video 10	Avg.
30	Meas.	16.33	7.69	7.32	6.77	9.26	11.53	13.69	17.06	14.29	15.99	11.99

	Est.	4.25	4.55	4.95	5.54	3.10	0.89	1.46	4.50	2.02	3.95	**3.52**
10	Meas.	15.66	8.07	7.74	7.15	9.53	12.34	13.70	17.32	14.82	16.70	12.30
	Est.	3.59	4.15	4.51	5.15	2.82	0.08	1.47	4.79	2.57	4.63	**3.38**
3	Meas.	19.73	9.49	10.43	8.28	11.45	12.92	13.90	18.04	13.75	16.47	13.45
	Est.	7.65	2.71	1.81	4.00	0.91	0.51	1.69	5.50	1.53	4.34	**3.07**
1	Meas.	21.89	14.33	15.92	13.58	15.57	18.19	17.83	19.18	16.08	19.46	17.20
	Est.	8.16	0.40	1.96	0.39	1.47	4.03	3.92	4.85	2.12	5.68	**3.30**

Table 4. Distance Errors to Point C on Road 2.

FPS	Type	Video 1	Video 2	Video 3	Video 4	Video 5	Video 6	Video 7	Video 8	Video 9	Video 10	Avg.
30	Meas.	11.96	4.60	5.47	0.35	0.46	3.94	4.84	5.10	4.21	11.96	5.29
	Est.	7.04	1.39	0.69	0.22	0.31	0.71	1.66	1.78	1.22	8.83	**2.39**
10	Meas.	12.00	5.37	5.55	0.87	0.74	3.56	4.39	4.65	4.22	11.16	5.25
	Est.	7.02	1.99	0.63	0.70	0.57	0.33	1.22	1.33	1.20	8.03	**2.30**
3	Meas.	14.32	6.09	5.89	2.23	1.91	2.30	2.48	3.09	2.63	8.56	4.95
	Est.	9.31	1.58	0.92	2.06	1.76	0.93	0.71	0.22	0.46	5.40	**2.33**
1	Meas.	17.78	8.45	11.18	5.73	4.31	1.15	1.26	1.46	1.41	5.18	5.79
	Est.	6.33	2.59	0.22	0.49	0.89	0.72	0.60	0.46	0.44	7.01	**1.97**

Figure 8(a) shows the measured and estimated speeds of Video1 of Road 1 when the frame matching speed is 30 FPS. Figures 8(b) and 8(c) show the biases in the x and y directions, and the actual, measured, and estimated distances to Points A, B, and C. Figures 9–11 show the cases where the frame matching speeds are 10, 3, and 1 FPS, respectively. Figures 12–15 show the same cases of video 1 of Road 2.
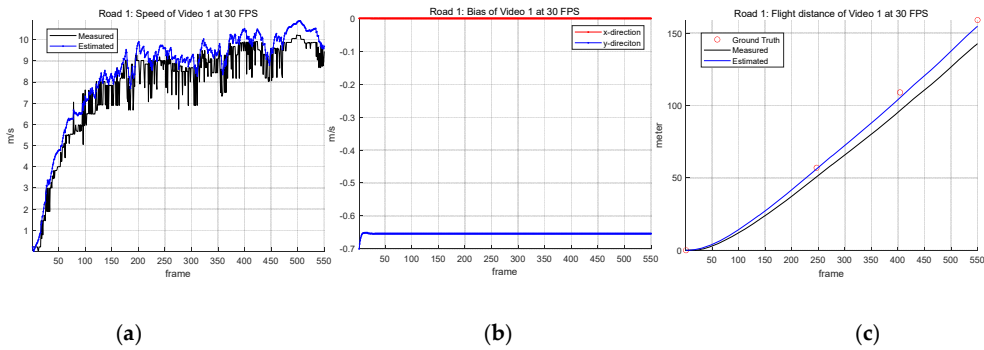

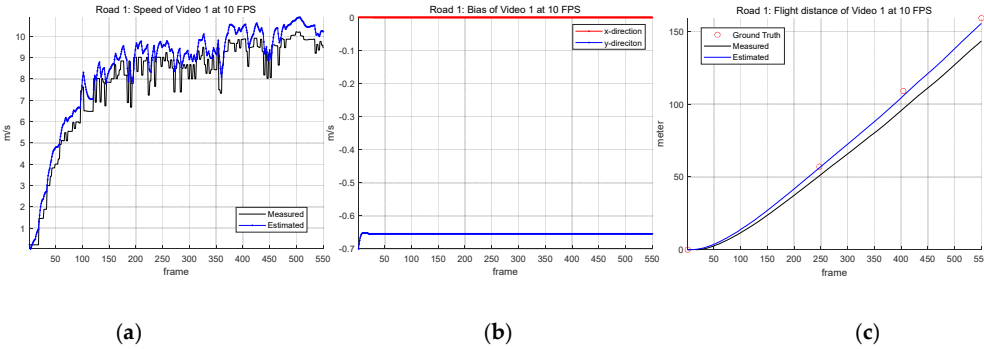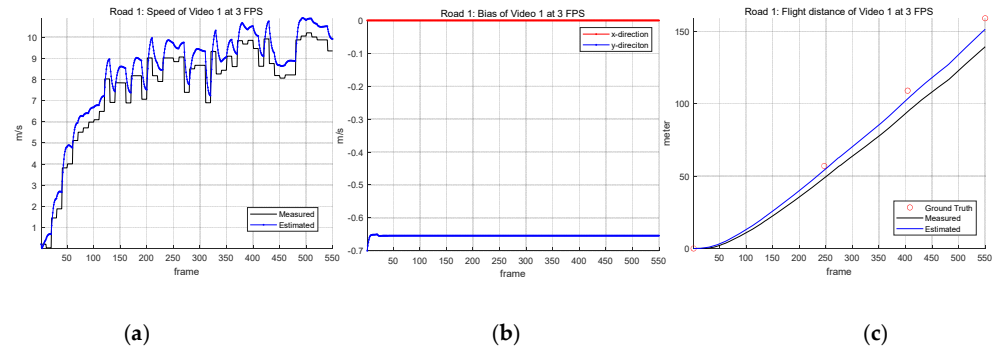**Figure 8.** Road 1: Video 1 with 30 FPS frame matching speed, (**a**) speed, (**b**) bias, (**c**) flight distance.
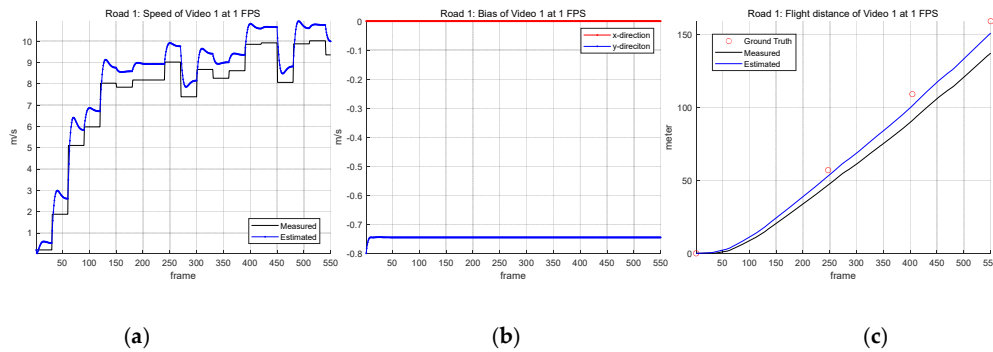

Figure 9. Road 1: Video 1 with 10 FPS frame matching speed, (**a**) speed, (**b**) bias, (**c**) flight distance.
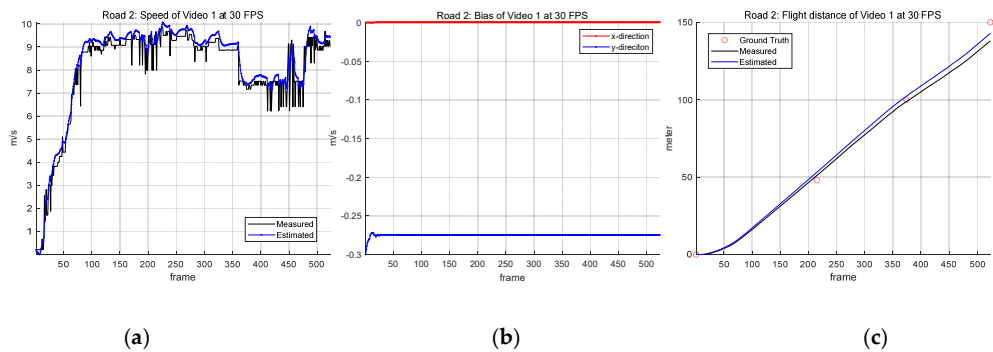
Figure 10. Road 1: Video 1 with 3 FPS frame matching speed, (**a**) speed, (**b**) bias, (**c**) flight distance.

Figure 11. Road 1: Video 1 with 1 FPS frame matching speed, (**a**) speed, (**b**) bias, (**c**) flight distance.

Figure 12. Road 2: Video 1 with 30 FPS frame matching speed, (**a**) speed, (**b**) bias, (**c**) flight distance.
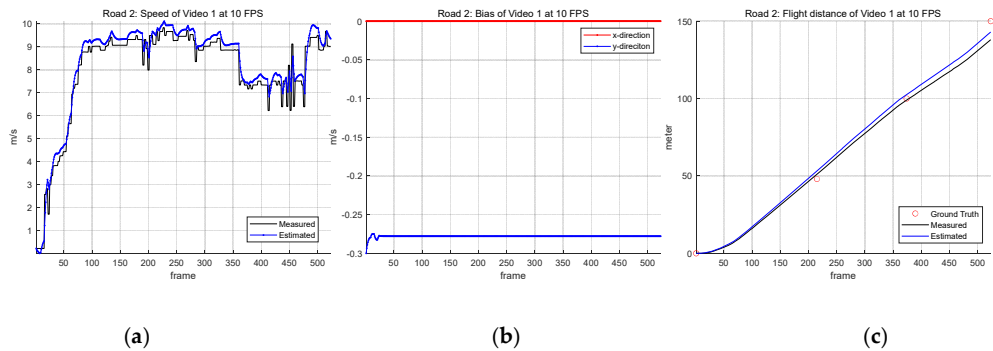

Figure 13. Road 2: Video 1 with 10 FPS frame matching speed, (**a**) speed, (**b**) bias, (**c**) flight distance.
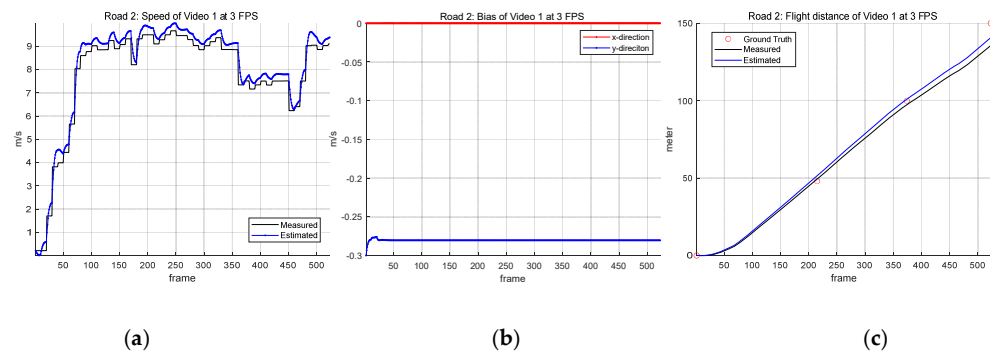
Figure 14. Road 2: Video 1 with 3 FPS frame matching speed, (**a**) speed, (**b**) bias, (**c**) flight distance.
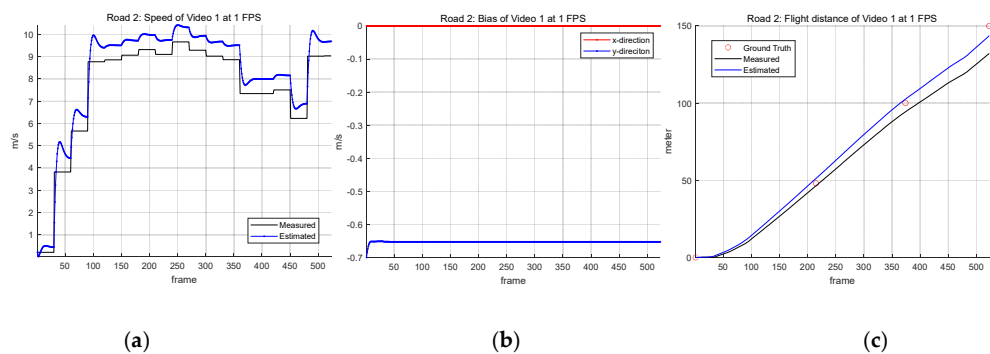

Figure 15. Road 2: Video 1 with 1 FPS frame matching speed, (**a**) speed, (**b**) bias, (**c**) flight distance.

Tables 5 and 6 show the average distance errors to Points A, B, and C for Road 1 and Road 2, respectively. Table 5 shows a regular pattern, with the error increasing as the distance increases and the frame matching speed decreases. However, Table 6 shows a somewhat irregular pattern in the average error in the estimated velocity as a function of distance or matching speed. This is due to different initial bias values which affect the performance.

Table 5. Average Distance Errors to Points A, B, C on Road 1.

FPS	Type	Point A (57 m)	Point B (109 m)	Point C (159 m)
30	Meas.	5.17	9.58	11.99
	Est.	**1.68**	**2.35**	**3.52**
10	Meas.	5.47	9.93	12.30
	Est.	**1.68**	**2.34**	**3.38**
3	Meas.	6.52	11.24	13.45
	Est.	**1.51**	**2.39**	**3.07**
1	Meas.	10.16	15.12	17.20
	Est.	**3.74**	**4.81**	**3.30**

Table 6. Average Distance Errors to Points A, B, C on Road 2.

FPS	Type	Point A (48 m)	Point B (100 m)	Point C (150 m)
30	Meas.	3.25	4.27	5.29
	Est.	**3.18**	**3.30**	**2.39**
10	Meas.	2.88	4.07	5.25
	Est.	**2.83**	**3.00**	**2.30**
3	Meas.	1.99	3.29	4.95
	Est.	**1.84**	**1.69**	**2.33**

1	Meas.	2.41	4.23	5.79
	Est.	**1.34**	**2.19**	**1.97**

Twenty supplementary files are movies in MP4 format, and they are available online. Supplemental material videos S1–S10 show 10 videos capturing Road 1, and Supplemental material videos S11–S20 show another 10 videos capturing Road 2. The optical center of the camera is marked as '+' in blue color. As the drone passes Points A, B, and C, the color of the mark changes to white.

4. Discussion

The optimal windows aim to achieve uniform spacing in the real-world coordinates. The number of windows was intuitively predetermined. The measured displacements in all templates are equally weighted as in equation (2), but further research on adaptive weighting is needed.

The augmented-state Kalman filter with the NCA model improved the flight range accuracy from high to low meter-level error. The zero-order hold scheme provides similar accuracy regardless of the frame matching speed

It turns out that the initial bias setting is important. The suburban road can estimate the flight distance more accurately, but it is easily affected by the initial bias. Therefore, 10 videos of Road 2 were divided into 3 groups, and different initial biases in the y direction were applied to each group. If the frame matching speed were slower, the initial bias should be lowered. They were chosen heuristically when better results were produced. Adaptively selecting the initial bias depending on the scene complexity and dynamics also remains a future topic.

5. Conclusions

A novel frame-to-frame template matching method is proposed. The optimal windows are derived from ray optics principles and a piecewise linear regression model. Multiple templates are obtained by their corresponding optimal windows. Therefore, the templates are scene or object independent and no additional processes are required for template selection and update.

The Kalman filter adopts the NCA model and its state is augmented to estimate the velocity bias of the drone. The zero-order hold method was applied for faster processing. The proposed technique achieves low average flight distance errors even at slow frame matching speeds.

This technique can be useful when external infrastructure is not available such as GPS-denied environments. It could be applied to a variety of fields, including automatic programmed flight or multi ground target tracking using flying drones, which remains a subject of future research.

References

1. Z. Zaheer, A. Usmani, E. Khan and M. A. Qadeer, "Aerial surveillance system using UAV," 2016 Thirteenth International Conference on Wireless and Optical Communications Networks (WOCN), Hyderabad, India, 2016, pp. 1-7, doi: 10.1109/WOCN.2016.7759885.

2. Vohra, D.; Garg, P.; Ghosh, S. (2023). Usage of UAVs/Drones Based on Their Categorisation: A Review. Journal of Aerospace Science and Technology, 74, 90–101. https://doi.org/10.1007/s42496-023-00123-1.

3. Osmani, K.; Schulz, D. Comprehensive Investigation of Unmanned Aerial Vehicles (UAVs): An In-Depth Analysis of Avionics Systems. Sensors 2024, 24, 3064.

4. Würbel, Heike. (2017). Framework for the evaluation of cost-effectiveness of drone use for the last-mile delivery of vaccines.

5. Zhang, Z.; Zhu, L. A Review on Unmanned Aerial Vehicle Remote Sensing: Platforms, Sensors, Data Processing Methods, and Applications. Drones 2023, 7, 398. https://doi.org/10.3390/drones7060398.

6. Mohammed, F., Idries, A., Mohamed, N., Al-Jaroodi, J., and Jawhar, I. (2014). UAVs for smart cities: Opportunities and challenges. Future Generation Computer Systems, 93, 880–893. https://doi.org/10.1016/j.future.2018.10.011.

7. Chen, C.; Tian, Y.; Lin, L.; Chen, S.; Li, H.; Wang, Y.; Su, K. Obtaining World Coordinate Information of UAV in GNSS Denied Environments. Sensors 2020, 20, 2241. https://doi.org/10.3390/s20082241.

8. Mokhamad Nur Cahyadi, Tahiyatul Asfihani, Ronny Mardiyanto, Risa Erfianti, Performance of GPS and IMU sensor fusion using unscented Kalman filter for precise i-Boat navigation in infinite wide waters, Geodesy and Geodynamics, Volume 14, Issue 3, 2023, Pages 265-274, https://doi.org/10.1016/j.geog.2022.11.005.

9. Kovanič, Ľ.; Topitzer, B.; Peťovský, P.; Blišťan, P.; Gergeľová, M.B.; Blišťanová, M. Review of Photogrammetric and Lidar Applications of UAV. Appl. Sci. 2023, 13, 6732. https://doi.org/10.3390/app13116732.

10. Petrlik, M.; Spurny, V.; Vonasek, V.; Faigl, J.; Preucil, L. LiDAR-Based Stabilization, Navigation and Localization for UAVs. In Proceedings of the 2021 International Conference on Unmanned Aircraft Systems (ICUAS), Athens, Greece, 15–18 June 2021; pp. 1220–1229. Available online: https://aerial-core.eu/wp-content/uploads/2021/11/ICUAS_2021_Matej.pdf (accessed on 5 May 2025).

11. Gaigalas, J.; Perkauskas, L.; Gricius, H.; Kanapickas, T.; Kriščiūnas, A. A Framework for Autonomous UAV Navigation Based on Monocular Depth Estimation. Drones 2025, 9, 236. https://doi.org/10.3390/drones9040236.

12. Yingxiu Chang, Yongqiang Cheng, Umar Manzoor, John Murray, A review of UAV autonomous navigation in GPS-denied environments, Robotics and Autonomous Systems, Volume 170, 2023, 104533, https://doi.org/10.1016/j.robot.2023.104533.

13. Yang, B.; Yang, E. A Survey on Radio Frequency Based Precise Localisation Technology for UAV in GPS-Denied Environment. Journal of Intelligent and Robotic Systems 2021, 101, 35.

14. Jarraya, I., Al-Batati, A., Kadri, M.B. et al. Gnss-denied unmanned aerial vehicle navigation: analyzing computational complexity, sensor fusion, and localization methodologies. Satell Navig 6, 9 (2025). https://doi.org/10.1186/s43020-025-00162-z.

15. R. C. Gonzalez and R. E. Woods, Digital Image Processing, 4th ed., Pearson, 2018.

16. R. Brunelli, "Template matching techniques in computer vision: a survey," Pattern Recognition, vol. 38, no. 11, pp. 2011–2040, 2003. DOI: 10.1016/j.patcog.2004.03.011.

17. D. Scaramuzza and F. Fraundorfer, "Visual Odometry [Tutorial]," IEEE Robotics and Automation Magazine, vol. 18, no. 4, pp. 80–92, 2011.

18. Stauffer, C., and Grimson, W. E. L. (1999). Adaptive background mixture models for real-time tracking. In CVPR, vol. 2, pp. 246–252.

19. Forster, C., Pizzoli, M., and Scaramuzza, D. (2014). SVO: Fast semi-direct monocular visual odometry. IEEE International Conference on Robotics and Automation (ICRA), pp. 15–22., https://doi.org/10.1109/ICRA.2014.6906584.

20. Kalal, Z., Mikolajczyk, K., and Matas, J. (2012). Tracking-Learning-Detection. IEEE Transactions on Pattern Analysis and Machine Intelligence, 34(7), 1409–1422. https://doi.org/10.1109/TPAMI.2011.239.

21. E. Hecht. "Optics" 5nd edition (Pearson, 2017).

22. Yeom, S. Long Distance Ground Target Tracking with Aerial Image-to-Position Conversion and Improved Track Association. Drones 2022, 6, 55.

23. Yeom, S.; Nam, D.-H. Moving Vehicle Tracking with a Moving Drone Based on Track Association. Appl. Sci. 2021, 11, 4046.

24. Muggeo, V. M. R. (2003). *Estimating regression models with unknown break-points.* Statistics in Medicine, 22(19), 3055–3071. DOI: 10.1002/sim.1545.

25. Bishop, Pattern Recognition and Machine Learning, Springer.

26. OpenCV Developers. (2024). Template Matching. OpenCV.org. https://docs.opencv.org/4.x/d4/dc6/tutorial_py_template_matching.html.

27. Bar-Shalom, Y.; Li, X.R. Multitarget-Multisensor Tracking: Principles and Techniques; YBS Publishing: Storrs, CT, USA, 1995.

28. Simon, D. (2006). *Optimal state estimation: Kalman, H∞, and nonlinear approaches*. Wiley-Interscience.

29. Anderson, B. D. O., and Moore, J. B. (1979). Optimal Filtering. Prentice-Hall.

30. https://dl.djicdn.com/downloads/DJI_Mini_4K/DJI_Mini_4K_User_Manual_v1.0_EN.pdf.

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.